

PUBLICATIONS

G.N. Ramachandran

Mathematical Philosophy Group
Indian Institute of Science
Bangalore 560 012

VOLUME - V

MR 33 - 38

BOOLEAN ALGEBRA APPLIED TO STATEMENTS OF TYPE 1 IN
QUANTIFIED PREDICATE LOGIC (QPL) AND ITS CONSEQUENCES
FOR CATEGORICAL SYLLOGISMS OF TRADITIONAL LOGIC

G.N. Ramachandran
Gita, 5, 10A Main Road
Malleswaram West
Bangalore 560 055

(For private circulation only)

Matphil Reports No. 33, December 1983

.i.

PREFACE

This report is different from the earlier reports in that it is not a paper written on work that has been completed before being taken for writing up. The sections 1 and 2 were completed but when the paper was written to report the results of these studies, several new possibilities came up, such as the discovery of the new connective "into" (\oplus) and the effect of this on the working out of the categorical syllogism of traditional logic. Therefore, an analysis of all possible 64 combinations was taken and an algebraic technique for working out the valid syllogisms in all the four figures was developed. When this was done, it was noticed that in addition to 18 categorical syllogisms having $(\underline{s} \Rightarrow \underline{p})$ and $(\underline{s} \Rightarrow \neg \underline{p})$, 18 more could be derived having $(\neg s \Rightarrow p)$ and $(\neg s \Rightarrow \neg p)$. These were therefore carefully checked and written up in the form of Table 6. Thereafter an attempt was made to obtain algorithms for working out an

individual statement of QPL with quantified inputs and outputs which showed that the process of working out a sequence of Type 1 statements of QPL was not readily possible. It appeared as if such statements occurring one after the other will have to be all taken together and an equivalent QPL statement from input to final output will have to be obtained by checking the reversals of everyone of the statements. This is a process which requires further studies and the present report was stopped at that level.

Although this report does not explain everything, several very interesting new results have been obtained and it gives a picture of how the studies developed and newer results were obtained. It is proposed to write a condensed version of this as Part III of our series "Boolean Vector-Matrix Representation of Extended Predicate Logic (EPL)". In view of this, this report has not been thoroughly corrected for uniformity of presentation, and notation, nomenclature, etc. All these changed by virtue of the needs for the further development of the subject, and therefore the manuscript is left as it is for this particular report.

Abstract

This report deals with the algebraic representation of QPL sentences and connectives which occur in the categorical syllogisms of traditional logic. Using only two quantifier states in statements of the type $(\forall x) (\underline{a}(x) \Rightarrow \underline{b}(x))$ and $(\exists x) (\underline{a}(x) \Rightarrow \underline{b}(x))$ (which have been named "Type 1 statements"), it is shown that the 64 possible moods of categorical syllogisms can be analysed and the valid ones deduced, by using only a small number of rules. These consist of (a) the equivalences between reversed statements, namely $(\forall x) (\underline{a} \Rightarrow \underline{b}) \equiv (\forall x) (\neg \underline{b} \Rightarrow \neg \underline{a})$, and $(\exists x) (\underline{a} \Rightarrow \underline{b}) \equiv (\exists x) (\underline{b} \Rightarrow \underline{a})$ (the latter replacing $(\exists x) (\underline{a} \& \underline{b})$ commonly used), (b) the effect of putting a QPL term as input "into" another quantified sentence (the newly introduced "into" operator (\otimes) leading to $\forall \otimes A = A$, $A \otimes \exists = \exists = \exists \otimes A$, $\exists \otimes \exists = \Delta$ (the indeterminate state of EPL)), and (c) converting a \forall statement, when needed, into an \exists statement by the QPL implication $(\forall x) (\underline{a}(x) \Rightarrow \underline{b}(x)) \supset (\exists x) (\underline{a}(x) \Rightarrow \underline{b}(x))$. Eighteen of the nineteen listed moods in the four figures of traditional logic

can be proved to be valid in this way, provided the conclusion

for $s \rightarrow m$, $m \rightarrow p$ can only be $(\forall x) (\underline{s} \Rightarrow \underline{p})$ or
 $(\forall x) (\underline{s} \Rightarrow \neg \underline{p})$, $\forall \equiv \forall$, or } .

Eighteen additional new moods can be shown to be valid,

which have the conclusions $(\forall x) (\neg \underline{s} \Rightarrow \underline{p})$ or $(\forall x) (\neg \underline{s} \Rightarrow \neg \underline{p})$

These include one that is superior in power to the mood Bamalip

in the fourth figure listed in the literature. It is also proved

that the rest of the 64 syllogisms lead to only the QPL state Δ ,

or the SNS state D, for the conclusion, and are therefore not valid.

Finally, it is shown that a sequence of QPL statements
of Type 1 cannot be worked out step by step, using the connective
 \Rightarrow , but has to be taken as a whole and processed. The
consequences of this will be reported later.

Contents

	Page No.
1. Introduction	1
2. Traditional Categorical Syllogisms in our Formalism	5B
(a) Use only of the quantifier states \forall and \exists for QPL.	7
(b) Application of the formulae in Section 2(a) in particular examples.	12
(c) Nature of the connective "into" for all pairs of quantifiers in EPL	15C
3. Theory of the new Connective "into" (\otimes) in EPL	
(a) Effect of \otimes applied between two basic states.	19
(b) Effect of "into" applied to all eight quantifier states of EPL	21
(c) Examples of syllogisms in Traditional Logic	23
4. Analysis of all possible 64 combinations in categorical syllogisms	28
(a) Sixteen possible combinations in the first figure	31
(b) Discussion of the second figure and reversal of QPL statements.	37
(c) Details of the applications for the second figure.	41
(d) Application to syllogisms in the third figure	43
(e) Application to the syllogisms in the fourth figure.	47
5. New type of syllogism leading to conclusions of the form $\neg \underline{s} \Rightarrow \underline{p}$ and $\neg \underline{s} \Rightarrow \neg \underline{p}$	
(a) Outline of the algebraic technique for working out valid syllogisms	49
(b) Proof of uniqueness of the technique	51

1. Introduction

In the earlier Parts I and II ([1], [2]), the basic theory of the vector-matrix representation of Boolean algebras was discussed and the application of BA-3 to represent the different possible states in quantified predicate logic was considered. It was shown that BA-3 can represent the four well-known states of standard quantified predicate logic (QPL), namely "for all" (\forall), "for none" (\emptyset), "there exists" (\exists), "not for all" (\nexists), in the form of Boolean 3-vectors, namely $(1 \ 0 \ 0)$, $(0 \ 0 \ 1)$, $(1 \ 1 \ 0)$, $(0 \ 1 \ 1)$ respectively. Since BA-3 has four additional states to make up the full complement of $2^3 (=8)$ possible different vectors, the new states were also defined and given the names "some" (Σ) = $(0 \ 1 \ 0)$, "all or none" (Θ) = $(1 \ 0 \ 1)$, "indefinite" (Δ) = $(1 \ 1 \ 1)$ and "impossible" (ϕ) = $(0 \ 0 \ 0)$. It was shown that the standard form of QPL can be extended readily to include the eight states, that has been term "Extended Predicate Logic" (EPL) and that the different standard forms of

.2.

expressing a state in QPL by six Boolean elements (See Table 4, Part I) can be converted to canonical forms which are representable by 3-element BA-3 vectors.

In Part II, the 3×3 Boolean matrix connectives in QPL were discussed and, in particular, the analogues of "and" and "or" in classical propositional calculus and SNS logic were defined. These matrix connectives relate one EPL state for one variable x (e.g.

$(\forall x) (\underline{a}(x))$ with another EPL state in a variable y (e.g. $(\exists y) (\underline{b}(y)$)

Such a connective ($\underline{\lambda}$, in general) is a pure EPL connective and

there are $8 \times 8 = 64$ of the type \underline{A} ("and") and \underline{O} ("or")⁶⁴

An example of the application of these connectives in the form of

a practical problem was given in Section 3 of Part II. In general,

the formalism developed in Part II deals with, what may be termed, EPL sentences of Type 2, which have the form of Eq.(1) for a unary relation.

$$(\forall_1 x) (\underline{a}(x)) \underline{\lambda} = (\forall_2 y) (\underline{b}(y)) \quad (1)$$

On making further studies on the subject and trying to prove the well-known results in predicate logic, such as the different figures and moods of traditional logic, it was found that these relations had a different syntax from the ones considered in Parts I and II. They are all expressed for a single variable (x) only, and connect different statements inside the quantifier (with one another). For example,

$$(\forall x) (\underline{b}(x) \Rightarrow \underline{c}(x)) , (\forall x) (\underline{a}(x) \Rightarrow \underline{b}(x)) \\ \equiv (\forall x) (\underline{a}(x) \Rightarrow \underline{c}(x)) \quad (\text{Barbara}) \quad (2a)$$

$$(\forall x) (\underline{b}(x) \Rightarrow \underline{c}(x)) , (\exists x) (\underline{a}(x) \& \underline{b}(x)) \\ \equiv (\exists x) (\underline{a}(x) \& \underline{c}(x)) \quad (\text{Darii}) \quad (2b)$$

In these, $(\forall x) (\underline{a}(x) \Rightarrow \underline{b}(x))$ stands for "All \underline{a} are \underline{b} " and $(\exists x) (\underline{b}(x) \& \underline{c}(x))$ stands for "Some \underline{a} are \underline{b} " or "There exist \underline{a} 's that are \underline{b} ", which employ the two quantifiers $(\forall x)$ and $(\exists x)$ respectively. The general form of this equation which represent all equations of this type may be expressed as below, in (2c) or (2d).

$$(\forall_1 x) (\underline{a} z_1 = \underline{b}), (\forall_2 x) (\underline{b} z_2 = \underline{c}) \equiv (\forall_3 x) (\underline{a} z_3 = \underline{c}) \quad (2c)$$

→

Equations of this type, in which only SNS connectives occur, and all quantifiers are for the same variable (x say) will be named "Type 1 equations" of QPL, or EPL. The rules governing the

$$(\forall_1 x) (\underline{a}_1), (\forall_2 x) (\underline{a}_2(x)) \equiv (\forall_3 x) (\underline{a}_3) \quad (2d)$$

formation of the right hand side in equations of this type were examined and the main results that were obtained are given below in this paper. In Sections 2 and 3, we shall first consider generally equations of Type 1, in which Z_1 , Z_2 , Z_3 are all SNS connectives (particularly implicants) and the nature of \mathcal{Q}_3 and the nature of the connective Z_3 are considered (given \mathcal{Q}_1 , \mathcal{Q}_2 , Z_1 , Z_2). The relevance of these to the well-known results in the list of equations of the type 1 in traditional logic are then briefly considered, and some examples are given to indicate their nature and validity.

The study was made of the Boolean algebraic rules that have to be applied to get the output sentence which is the resultant of two input sentences as found in categorical syllogisms. It was possible to show that 18, out of the 19 that are listed in books, could be derived as a result of successive implications. 18 more were obtained in which the implication

in the conclusion starts from $\neg \underline{s}$ towards \underline{p} or $\neg \underline{p}$.

This is a very interesting consequence and it is discussed particularly in Section 5.

On the other hand, quantified sentential relations (named Type 2 QPL equations) for two variables, of the type considered in Parts I and II — namely

$$(\forall_1 x) (\underline{a}(x)) \not\equiv (\forall_2 y) (\underline{b}(y)) \equiv \underline{c} \quad (3)$$

lead to an entirely different algebra, as described therein.

As an illustration, the interpretation of the quantifier $(\forall x)$

as "for all (collectively)" and "for each and every

(individually)" are not exactly equivalent and may have

different consequences in the types of equations discussed in

Part II (e.g. Eq (3)), and the type of equation with one variable

throughout as shown in Eq (1). Similarly, the use of $(\exists x)$ in

.5A.

connecting two QPL sentences has also different possibilities,

such as its being transferable (transitive) for Type 2 equations,

The possibility of

and its leading to no consequences in the succeeding step —

i.e. to the QPL state (Δx) — for Type 1 equations.

In the succeeding sections of this paper, several formulae, or algorithms, are mentioned which will carry out a whole set of logical operations in QPL and/or EPL. It is not easy to give formal proofs for all these, because they have been arrived at as the extracted information from the logic that can be intuitively formulated from all the particular examples for which they are operative. In this sense, the method of approach is that of formulating a physical theory, where a number

. . 5B

dissected

of phenomena are ~~discussed~~ and examined and the common substratum behind all of them is formulated as a physical law.

After these studies are completed, it may be possible to give an axiomatic basis for all the formulations that have been made in connection with QPL and EPL. However, this is passed over in this paper and ^{an} ~~the~~ attempt is to derive empirical formulae which are simple, but which are valid over a wide range of circumstances. This should be remembered while checking the correctness and understanding the contents of the formulae given here.

2. Traditional Categorical Syllogisms in our Formalism

A brief outline of the list of categorical syllogisms of traditional logic, in various figures and moods, is given in the article by A. Church in Encyclopedia Britannica [3], p.216. We shall follow this presentation in our discussion, as it is elegant and is written in a format that is immediately adaptable to the form of a QPL statement as in Eqns (1a), (1b). On examining

the 19 syllogisms listed by Church, it is found that the only types of QPL statements that occur are

$$(\forall x) (\underline{a} \Rightarrow \underline{b}), \quad (\forall x) (\underline{a} \Rightarrow \neg \underline{b}) \quad (4a, b)$$

$$(\exists x) (\underline{a} \& \underline{b}), \quad (\exists x) (\underline{a} \& \neg \underline{b}) \quad (4c, d)$$

In the QPL terms 4(a to d), and also in what follows in this section, we shall put the variable (x) associated only with the quantifier (as in $(\forall x)$, $(\forall x)$ etc) and omit it in the SNS sentence that is quantified. Thus, it is understood that \underline{a} stands for $\underline{a}(x)$ and \underline{b} stands for $\underline{b}(x)$ etc, and the same variable x associated with the quantifier occurs also for the SNS term. Also, in all derivations of \underline{b} from \underline{a} , given the truth of the connective relation $\underline{a} \gtrsim \underline{b}$, this is obtained from the unary form of the logical relation $\underline{a} \gtrsim = \underline{b}$. This follows from the discussion in Part I [1] from which it is readily seen that,

$$\underline{a} \gtrsim \underline{b} = \underline{c}, \quad \underline{c} = T \rightarrow \underline{a} \gtrsim = \underline{b} \quad (5)$$

Therefore, in considering syllogisms having the major premise,

the minor premise and the conclusion, we need consider only the most general expression given by Eq (2), from which all particular cases of Type 1 equations can be derived by giving the nature of the QPL connectives $\mathcal{Q}V$ and the SNS connectives \mathcal{Z} .

In this section, we shall consider first the four standard states of QPL namely, \forall , \exists , \top , \perp and discuss the way they combine for a general equivalence of the type of Eq (2). More particularly given the logical states $(\mathcal{Q}V_1x)$, $(\mathcal{Q}V_2x)$ and \underline{a} we wish to derive the quantifier $(\mathcal{Q}V_3x)$ and the truth value of \underline{c} . We give below an algorithm for doing this. The proof is omitted as the algorithm is derived as a condensation of the logical properties of the terms and connectives employed in this equation and it is readily verified that the algorithm works, and is in accord with what is intuitively deduced in each case.

(a) Use only of the quantifier states \forall and \exists for QPL.

As shown in Table 4 of Part I, the four standard quantifier

states can all be expressed in terms of the one quantifier state

\forall alone provided two "not's" (\neg and $\overline{\neg}$) are used in addition,

[see also [4].]

in front of it, or after it. We shall not employ \neg before the

quantifier, as it does not lead to a general elegant formula.

Then, an examination of Table 6 of Part I [1] shows that the four

quantifier states of standard QPL can be expressed in terms of

\forall and \exists as follows.

$$\forall = (\forall x) (\underline{s}(x)) ; \exists = (\exists x) (\underline{s}(x)) ; \quad (6a)$$

$$\Phi = (\forall x) (\neg \underline{s}(x)) ; \Lambda = (\exists x) (\neg \underline{s}(x)) \quad (6b)$$

Hence we need consider only the two quantifier states $(\forall x)$ and

$(\exists x)$ and associate with them statements $(\underline{s}(x))$ and $(\neg \underline{s}(x))$

which are either affirmative or negative. We shall now work out

the expressions for $(\forall_3 x)$ and \underline{z}_3 given $(\forall_1 x)$, $(\forall_2 x)$,

\underline{z}_1 and \underline{z}_2 , in Eq (2c).

We shall do this, particularly for the type of statements that occur in categorical syllogism, as in Eqns (4a,b,c,d),

which stand, respectively for

$$4a \quad \forall(\underline{s}): \text{ All } \underline{a} \text{ are } \underline{b}, \quad 4b \quad \forall(\neg\underline{s}): \text{ All } \underline{a} \text{ are not-} \underline{b} \quad (7a)$$

$$4c \quad \exists(\underline{s}): \text{ Some } \underline{a} \text{ are } \underline{b}, \quad 4d \quad \exists(\neg\underline{s}): \text{ Some } \underline{a} \text{ are not-} \underline{b} \quad (7c)$$

For these, if $\underline{s}(x)$ stands for " $\underline{a}(x)$ is $\underline{b}(x)$ ", then $\neg\underline{s}(x)$ stands for " $\underline{a}(x)$ is not- $\underline{b}(x)$ ". Therefore, we need only use the small set of four entries given in Table 1 below to get the quantifier \forall_3 given \forall_1 and \forall_2 as \forall or \exists in Eq.(2c) and (d). This is carried out by using a new type of QPL connective, for which we have given the name "into" (see the sections below for its different applications).

Table 1. Short table for the connective "into" (\oplus) for use in QPL.

The explanation of the entries for \forall_3 in Table 1 is obvious.

When both \forall_1 and \forall_2 are $(\forall x)$, the sentences \underline{s}_1 and \underline{s}_2

quantified by them are valid "for all x" and therefore the

resultant sentence is also valid "for all x" and hence $\forall_3 = (\forall x)$

On the other hand, if one of the two sentences \underline{s}_1 or \underline{s}_2 is

Table 1. Short table for the connective
"into" (\otimes) for use in QPL*

\mathcal{V}_1	\mathcal{V}_2	\mathcal{V}_3
A	A	A
A	E	E
E	A	E
E	E	Δ

*The quantifier state \mathcal{V}_3 is as in
Eq (2), when \mathcal{V}_1 and \mathcal{V}_2 are given.

valid for $(\forall x)$, while the other is valid only for the state $(\exists x)$, then, when the output of the first sentence, for which it is valid for $(\forall x)$, enters as input into the second sentence, valid only for $(\exists x)$, then it makes the resultant output of the second valid only for $(\exists x)$. Similarly, if the first sentence has the quantifier $(\exists x)$ and the second $(\forall x)$, the $(\exists x)$ in the output of the first one restricts the output of the second to be only $(\exists x)$. It is to be noted that this behaviour is quite different from the situations considered in Part II, where the two sentences that are combined are for different variables x and y .

On the other hand, when the first sentence gives an output $\underline{b}(x)$ for $(\exists x)$ and this is fed into the second sentence which also has the quantifier $(\exists x)$ then there is no reason why the value of x for which 'there exist' data in the output of the first sentence and the values of x for which the data are taken in the second sentence need overlap. The two states may be disjoint, in which case the quantifier of \underline{s} becomes

Φ ("for none" = (0 0 1)). On the other hand, if it happens that they ~~are~~ overlap, then \underline{s}_3 will have the quantifier ($\exists x$) ("there exists" = (1 1 0)). Since both possibilities can happen, we get the state $\Delta = (1 1 1)$ for Ψ_3 in the fourth entry in Table 1. It is to be noted that in the case of two different variables x and y (in sentences of Type 2 considered in Part II) the quantifier of the output of the first sentence, namely 'there exists', is the same as the quantifier of the input in the next equation, and therefore, even if $\underline{b}(x)$ is true for one value of x, then it can be input into next equation, under the quantifier state \exists .

(b) Application of the formulae in Section 2(a) in particular examples.

While the general formulation given in Section 2(a) for QPL is readily seen to be valid quite generally for all \underline{Z} 's, since this report deals essentially with traditional categorical syllogisms, we shall illustrate a few of these and derive them from the general formulae given here. Thus, we have already seen two examples in Section 1 — namely Eq 1(a) and Eq 1(b) —

which refer to the moods Barbara and Darii in the first figure.

The first of these, namely Barbara, is immediately obvious and corresponds to the first entry in Table 1. Similarly, the second example, viz. Eq 1(b), corresponding to the mood Darii, follows from the third row in Table 1. Writing it in our notation, we have,

$$(\forall x) (\underline{\underline{b}} \underline{\underline{I}} = \underline{\underline{c}}) , (\exists x) (\underline{\underline{a}} \underline{\underline{A}} = \underline{\underline{b}}) \quad (8)$$

gives, in so far as the quantifier is concerned, the resultant state

$$(\exists) \otimes (\forall) = (\exists) \quad (9a)$$

For the statement inside the bracket quantified by $(\exists x)$, we have the SNS result $\underline{\underline{a}} \underline{\underline{A}} \underline{\underline{I}} = \underline{\underline{c}}$. Putting in the matrices for the SNS connectives $\underline{\underline{A}}$ and $\underline{\underline{I}}$, we have,

$$|A| I = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} = |A| \quad (9b)$$

Hence, combining the r.h.s. of (9a) and (9b), we have the resultant of the two quantified terms in (8) as in Eq (1b), namely

$$(\exists x) (\underline{\underline{a}} \underline{\underline{A}} = \underline{\underline{c}}) \quad (9c)$$

We shall give two more examples of traditional syllogisms, and prove them from our theory. Thus, in the second figure we have Cesare which has the following equations.

$$(\forall x) (\underline{a}(x) \Rightarrow \underline{b}(x)), (\forall x) (\underline{c}(x) \Rightarrow \neg \underline{b}(x)) \quad (10 \text{ a, b})$$

it follows that

$$(\forall x) (\underline{c}(x) \Rightarrow \neg \underline{a}(x)) \quad (10c)$$

The critical step in proving this is to write,

$$(\underline{a}(x) \Rightarrow \underline{b}(x)) \equiv (\underline{a} \perp = \underline{b}) \quad (11a)$$

and convert it into its equivalent contrapositive form

$$(\neg \underline{b}) \perp = (\neg \underline{a}) \quad (11b)$$

This follows, since $(\underline{a} \perp = \underline{b})$, $\underline{b} \perp^t = \underline{a}$ and $\perp^t = \perp = \perp \perp \perp \perp$.

Thus, (10 a, b) are reduced to two implications as in (11c),

and the output of the first is the input of the next:

$$(\forall x) (\underline{c} \Rightarrow \neg \underline{b}), (\forall x) (\neg \underline{b} \Rightarrow \neg \underline{a}) \quad (11c, d)$$

Hence, the net result, on combining (11c) and (11d) is (10c),

as required.

We shall now consider an example of the third figure, namely Darapti which is given in (18), following Church's way [3] of writing this syllogisms. Thus, given (12 a & b),

$$(\forall x) (\underline{a} \Rightarrow \underline{b}), (\forall x) (\underline{a} \Rightarrow \underline{c}) \quad (12a, b)$$

it follows that

$$(\exists x) (\underline{a}) \mapsto (\exists x) (\underline{b} \& \underline{c}) \quad (12c)$$

The important part of this is to input the information $(\exists x) (\underline{a})$ in Eq (12a) and Eq (12b) and to deduce therefrom that $(\exists x) (\underline{b} \& \underline{c})$

"and" $(\exists x) (\underline{c})$ are true. These are derived by using the third line of Table 1 in (12a) and (12b). Hence it follows that the expression in (12e), namely $(\exists x) (\underline{b} \& \underline{c})$, is true.

We have followed the standard representation of the two quantified statements, such as "All students are intelligent" and "There exist students who are intelligent", by the symbols $(\forall x) (\underline{a}(x) \Rightarrow \underline{b}(x))$ and $(\exists x) (\underline{a}(x) \& \underline{b}(x))$, where \underline{a} stands for the property of being a student and \underline{b} stands for the property of being intelligent. x represents the variable denoting that the entity considered belongs to the collection of persons for which the above quantified statements are made.

In what has been considered above, we have followed this terminology in our notation of QPL and SNS and worked out the consequences for some syllagisms. It should be noted however that $\underline{a} \& \underline{b}$, in which the connective $\underline{\Lambda}$ is represented by the ~~2x2~~ matrix $\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$, is much stronger than $\underline{a} \Rightarrow \underline{b}$ for which

the connective \tilde{I} is represented by the matrix $\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$. If

we follow the interpretation of these matrices as we have

proved generally from the theory of relations and for SNS

logic in particular, then, in both cases, "when \underline{a} is true \underline{b} is true"

and " \underline{b} cannot be false when \underline{a} is true." However, what happens

when \underline{a} is false is entirely different in the two cases. In the

case of \tilde{I} (implies), when \underline{a} is F, there is no information

for the output,

content and \underline{b} can be either true or false, or whatever it was

before the application of the information contained in the

statement involving \tilde{I} . On the other hand, if $\underline{a} \tilde{A} = \underline{b}$ is

used, then \underline{a} itself cannot be negative, and there is a contradiction

if \underline{a} is false is applied. In other words, even at the stage

of saying \underline{a} is true, we say that \underline{a} cannot be false (the full

implication of this consequence has to be considered in great

detail) and the application of \tilde{A} to statements of this type

cannot be readily substantiated[§]. On the other hand, the

objection to writing $\underline{a} \Rightarrow \underline{b}$ under the existential quantifier ($\exists x$)

that is commonly made (see [5]) — namely that a false statement can imply any statement whatsoever — is not relevant in our formalism, since a false statement implies no statement in our formalism (and not any statement whatsoever).

In view of this, we shall consider the reformulation of all standard syllogisms in traditional logic in terms of the following symbolic statements in QPL, using only the implication symbol \Rightarrow :

$$(\forall x) (\underline{a} \Rightarrow \underline{b}) \equiv \text{"All } \underline{a} \text{ are } \underline{b}" \quad (13a)$$

and

$$(\exists x) (\underline{a} \Rightarrow \underline{b}) \equiv \text{"Some } \underline{a} \text{ are } \underline{b}" \quad (13b)$$

The negations of these implications are

$$(\forall x) (\underline{a} \Rightarrow \neg \underline{b}) \equiv \text{"All } \underline{a} \text{ are not-} \underline{b}" \quad (13c)$$

$$(\exists x) (\underline{a} \Rightarrow \neg \underline{b}) \equiv \text{"Some } \underline{a} \text{ are not-} \underline{b}" \quad (13d)$$

The reverses of these, from \underline{b} to \underline{a} , are readily verified from Venn-diagrams and are as follows:

$$(\forall x) (\underline{a} \Rightarrow \underline{b}) \equiv (\forall x) (\neg \underline{b} \Rightarrow \neg \underline{a}) ;$$

$$(\forall x) (\underline{a} \Rightarrow \neg \underline{b}) \equiv (\forall x) (\underline{b} \Rightarrow \neg \underline{a}) \quad (13e, f)$$

$$(\exists x) (\underline{a} \Rightarrow \underline{b}) \equiv (\exists x) (\underline{b} \Rightarrow \underline{a}) ; \quad (13g, h)$$

$$(\exists x) (\underline{a} \Rightarrow \neg \underline{b}) \equiv (\exists x) (\neg \underline{b} \Rightarrow \underline{a})$$

The second set (13g, h) is particularly to be noted - namely that its nature is different from the former two (13e, f). This has profound consequences for the structure of predicate logic, made of sentences of type 1.

(c) Nature of the connective "into" for all pairs of quantifiers in EPL

We shall illustrate the nature of the connective "into" in QPL, by taking a simpler example than Eq (2). Consider the following

$$\text{Input : } (\nexists_1 x) (\underline{a}') \quad (14a)$$

$$\text{Into equation : } (\nexists_2 x) (\underline{a} \nexists = \underline{b}) \quad (14b)$$

$$\text{Output : } (\nexists_3 x) (\underline{b}'), \text{ where } \underline{a}' \nexists' = \underline{b}' \quad (14c)$$

Then, on inputting a different quantifier ($\mathcal{Q}_2 x$) for $\underline{a}(x)$

into (13a), we obtain (using the same considerations as in

Section 2(b) the output quantifier to be

$$(\mathcal{Q}_1 x) \oplus (\mathcal{Q}_2 x) = (\mathcal{Q}_3 x) \quad (14d)$$

Since (14) is a very general rule for \mathcal{Q}_3 , for equations as in (13), we shall give a general table for the effect of the operator "into" on \mathcal{Q}_1 when introduced into \mathcal{Q}_2 (Table 2).

Table 2. Multiplication table for the connective "into" (\oplus) when \mathcal{Q}_1 and \mathcal{Q}_2 are given.

The extension of this table to the cases when either \mathcal{Q}_1 , or \mathcal{Q}_2 , or both, are one of the four non-standard states of EPL, namely $\Sigma, \Theta, \Delta, \emptyset$, is not very obvious and we shall discuss this in the next Section 3. We give below, in Table 2(b), a listing of the sixteen possible values of $\mathcal{Q}_1 \oplus \mathcal{Q}_2$ when \mathcal{Q}_1 and \mathcal{Q}_2 take the four possible QPL states $\forall, \exists, \Phi, \Lambda$, the

Table 2: Multiplication table for the connective "into" (\oplus) when Q_1 and Q_2 are given

(a) Q_3 and Z' , given Q_1 , Q_2 , Z , when Q_1 and Q_2 are both one of the four standard quantifiers in QPL

Q_1	Q_2	\forall	\exists	\forall	\exists	Φ	Ψ	\forall	\exists
Q_1	Q_3	Z'	Z'	Q_3	Z'	Q_3	Z'	Q_3	Z'
\forall	\forall	\forall	\exists	\exists	\exists	Φ	Ψ	Λ	Σ
\exists	\exists	\exists	Δ	Δ	\exists^*	Λ	Σ	Δ	Σ^+
Φ	\forall	Σ	\exists	\exists	Σ	Φ	Σ	Λ	Σ
Λ	\exists	Σ	Δ	Δ	Σ^+	Λ	Σ	Δ	Σ^*

* May also be Σ ; + May also be Σ .

(b) Q_3 in the canonical form when Q_1 and Q_2 are also given in the canonical form for the same four states of QPL as in (a) above.

Q_1	Q_2	\forall	\exists	Φ	Ψ	Λ
Q_1		\forall	\exists	Φ	Ψ	Λ
\forall	\forall	\forall	\exists	\forall	\exists	\forall
\exists	\exists	\exists	Δ	\exists	Δ	Δ
Φ	Φ	Φ	Δ	Φ	Δ	Δ
Λ	Λ	Δ	Δ	Λ	Δ	Δ

* Z' is omitted as it is Σ for all entries.

The portion marked out in the box is the essential minimum, on which this table is based, and which is given in Table 1.

values of \mathcal{V}_3 being also given in the QPL form. This leads to a great convenience for it turns out that the operator \mathcal{Z}' in Table 2(a) can always be made equal to \mathcal{E} , and consequently the statement that is quantified, namely $(\underline{a} \mathcal{Z} = \underline{b})$ is unaltered. This Table 2(b) will therefore be very useful in working out problems in QPL. It is to be noted, however, that sometimes the output for \mathcal{V}_3 could be one of the non-QPL states, namely Δ of EPL. In such a case, the resultant statement has no informational value and the corresponding syllogism will not have the third component and is therefore considered to be invalid. It is very interesting that traditional logic has fully comprehended the complete vagueness of the SNS statement when an input with $\mathcal{V}_1 = \exists$, is introduced into a $\mathcal{V}_2 = \exists$. Consequently there is no example of this type in the list of 19 syllogisms found in traditional logic (see [6]).

3. Theory of the new Connective "into" (\oplus) in EPL

(a) Effect of \oplus applied between two basic states.

Tables 1 and 2 cover all cases that would be met with, in standard QPL, of the connective "into". The extension of

these to the $8 \times 8 = 64$ possibilities of $\mathcal{V}_1 \oplus \mathcal{V}_2$, in EPL, is

not difficult. We first find out the resultant, \mathcal{V}_3 , of

$\mathcal{V}_1 \oplus \mathcal{V}_2$, for the nine cases which involve the three basic

states, each, of \mathcal{V}_1 and \mathcal{V}_2 , and then employ these for the

remaining cases by using the distributive laws

$$\mathcal{V}_i \oplus (\mathcal{V}_j \oplus \mathcal{V}_k) = (\mathcal{V}_i \oplus \mathcal{V}_j) \oplus (\mathcal{V}_i \oplus \mathcal{V}_k) \quad (15a)$$

and

$$(\mathcal{V}_j \oplus \mathcal{V}_k) \oplus \mathcal{V}_l = (\mathcal{V}_j \oplus \mathcal{V}_l) \oplus (\mathcal{V}_k \oplus \mathcal{V}_l) \quad (15b)$$

The results of the operation \oplus from \mathcal{V}_1 into \mathcal{V}_2 can

be listed as follows:

$$V \oplus V = V, \quad \emptyset \oplus \emptyset = \emptyset, \quad \Sigma \oplus \Sigma = \Delta \quad (16 \text{ a,b,c})$$

$$V \oplus \emptyset = V, \quad \emptyset \oplus V = \emptyset, \quad (16 \text{ d.e})$$

$$\Sigma \oplus V = V \oplus \Sigma = \Sigma, \quad \Sigma \oplus \emptyset = \emptyset \oplus \Sigma = \Sigma \quad (16 \text{ f,g,h,k})$$

Of these, (16a), (16b), (16d) and (16e) are taken from Table 2(b),

where they have been listed from intuition. The derivation of (16 f to k) is obtained also from this table as follows. For $\sum \oplus V$, we start from $\exists \oplus V = \exists$ in Table 2(b). Writing \exists as $(V \oplus \sum)$, we have

$$(V \oplus \sum) \oplus V = (V \oplus V) + (\sum \oplus V) = V \oplus \sum \quad (17)$$

Since $V \oplus V = V$ from (16a), we obtain (16f) from (17). The products in (16 g,h,k) similarly follow.

We are now left with (16c), namely $\sum \oplus \sum = \Delta$. For this, we use $\exists \oplus \exists = \Delta$ of Table 2(b). Expanding both sides in terms of the basic states V and \sum , we have

$$\begin{aligned} \exists \oplus \exists &= (V \oplus \sum) \oplus (V \oplus \sum) \\ &= (V \oplus V) \oplus (V \oplus \sum) \oplus (\sum \oplus V) \oplus (\sum \oplus \sum) \\ &= V \oplus \sum \oplus \sum + (\sum \oplus \sum) = \Delta = V + \sum + \bar{\Phi} \quad (18) \end{aligned}$$

This shows that $\bar{\Phi}$ is one of the components of $(\sum \oplus \sum)$.

Similarly, by expanding $\Delta \oplus \Delta = \Delta$, we find that $\Sigma \oplus \Sigma$ contains \forall . Obviously, "some" "into" "some" can be "some" itself, so that $\Sigma \oplus \Sigma$ contains Σ also. Hence,

$$\Sigma \oplus \Sigma = \forall + \emptyset + \Sigma = \Delta = (1 \ 1 \ 1) \quad (19)$$

[As a check, we can work back $\exists \oplus \exists$, and the three related products, and we find that they are all equal to Δ , as listed in Table 2(b)].

For ready reference, Table 3 gives the products obtained via the "into" operator ~~from one of~~ ^{from one of} ~~for~~ ^{to} the three basic EPL states ~~with one~~ another.

Table 3: Multiplication table of $\mathcal{Q}_1 \oplus \mathcal{Q}_2$ for the three basic states.

(b) Effect of "into" applied to all ~~eight~~^(quantifier) states of EPL

Table 4 giving this was obtained by expanding each term as the Boolean sum (\oplus) of basic states, and using (15a and b) and Table 3. All entries are in the canonical form.

Table 3: Multiplication table of $\Psi_1 \oplus \Psi_2$
for the three basic states

Φ_1	Φ_2	A	Σ	Δ	M	A
Σ	Δ	Δ	Σ	Δ	Σ	Δ
Φ_1	Φ_2	Σ	Δ	Σ	Δ	Σ
A	A	A	A	A	A	A

Table 4: 8 X 8 multiplication table for the connective "into"

(c) Examples of syllogisms in Traditional Logic

(i) Baroco: We shall illustrate the use of Table 2(b) in deriving the syllogisms, by first considering the mood Baroco of the second figure in Traditional Logic. The syllogism reads as follows (as per Ref [3]) (see also Section 2 above for the notation in our formalism which we use).

$$(\forall x) (\underline{\underline{p}}(x) \Rightarrow \underline{\underline{m}}(x)) , (\exists x) (\underline{\underline{s}}(x) \Rightarrow \neg \underline{\underline{m}}(x)) \quad (20a, b)$$

Hence $(\exists x) (\underline{\underline{s}}(x) \Rightarrow \neg \underline{\underline{p}}(x)) \quad (20c)$

Firstly, from SNS, using $(\underline{\underline{a}} \Rightarrow \underline{\underline{b}}) \equiv (\neg \underline{\underline{b}} \Rightarrow \neg \underline{\underline{a}})$, we can rewrite (20a) as

$$(\forall x) (\neg \underline{\underline{m}}(x) \Rightarrow \neg \underline{\underline{p}}(x)) \quad (20d)$$

Then, the SNS output of (20b) is the input in (20d). Also, the quantifier of the output of (20b), namely \exists , going "into" the quantifier \forall of (20d), yields $\exists \otimes \forall = \exists$. Hence, (20b) and (20d) together yield a statement having the quantifier \exists , and the SNS statement $\underline{\underline{s}} \Rightarrow \neg \underline{\underline{p}}$, which yields the required result (20c).

(ii) Darii: The example Darii, of the first figure, rewritten in our formalism, is as follows:

$$\equiv (\exists x) (\underline{s}(x) \Rightarrow \underline{p}(x)) \quad (21c)$$

The proof follows in a straightforward manner by applying (21b) into (21a), when the resultant quantifier is $\exists \oplus \forall = \exists$ and the resultant SNS connective is $\tilde{\wedge} \tilde{\wedge} = \tilde{\wedge}$, thus yielding (21c).

(iii) Darapti: In the example Darapti, considered in Section 2(b), the conclusion is given as $(\exists x) (\underline{b} \& \underline{c})$. However, this implies $(\exists x) (\underline{b} \Rightarrow \underline{c})$, so that the syllogism can also be written, in our formalism, in the form

$$(\forall x) (\underline{m}(x) \Rightarrow \underline{p}(x)) , (\forall x) (\underline{m}(x) \Rightarrow \underline{s}(x)) , (\exists x) (\underline{m}(x)) \quad (22a,b,c)$$

$$\equiv (\exists x) (\underline{s}(x) \Rightarrow \underline{p}(x)) \quad (22d)$$

In fact, we can go further, and write, as the consequence of

(22a, b, c), the result

$$(\exists x) (\underline{\underline{p}}(x) \Rightarrow \underline{\underline{s}}(x)), \text{ or } (\exists x) (\underline{\underline{s}}(x) \Leftarrow \underline{\underline{p}}(x)) \quad (22e)$$

and, it also follows, from (22d) and (22e), that (22a, b) lead to

$$(\exists x) (\underline{\underline{p}}(x) \Leftrightarrow \underline{\underline{s}}(x)) \quad (22f)$$

It appears that (22f) is the most powerful conclusion that can be derived from (22a,b,c) (see section 4(b) and Eq (28)).

(iv) Bamalip: This example, of the fourth figure, is interesting since the connective "into" is applied twice. It reads as follows (as per Ref [3]):

$$(\forall x) (\underline{\underline{p}}(x) \Rightarrow \underline{\underline{m}}(x)), (\forall x) (\underline{\underline{m}}(x) \Rightarrow \underline{\underline{s}}(x)), (\exists x) (\underline{\underline{p}}(x)) \quad (23a, b, c)$$

lead to

$$(\exists x) (\underline{\underline{p}}(x) \Rightarrow \underline{\underline{s}}(x)) \quad (23d)$$

Taking (23c) and applying its quantifier "into" that of (23a), we get the output $(\exists x) (\underline{\underline{m}}(x))$. Applying this once again "into" (23b), we get the output $(\exists x) (\underline{\underline{s}}(x))$. Thus, we obtain effectively the statement (23d). The resultant (23d) is not the most powerful one to be obtained from (23a, b) (see section 5).

It has been possible to check that, with the notation adopted by us, all the examples of the classical syllogisms given in Ref [3] could be proved simply, as indicated for the small number of examples shown above. On the other hand, since each syllogism has three members in it, each of which can have four forms, and two of them are inputs, there can be 16 possible syllogisms for each of the four figures, leading to 64 possible examples, as mentioned in Ref [6] page 20, 21. Of these, only 19 represent valid arguments according to the classical Aristotelian framework. It would be worthwhile to consider why most of them are invalid. It can arise from one of the two possibilities.

(a) The connection of the quantifiers in the first two members might lead, via the connective "into" (Θ), to the "indefinite" state Δ .

or (b) The combination of the implications may be such that

the output of the first statement is the negation of the input of the second statement, so that the resultant of the SNS implication of the two statements will together lead to the "doubtful" state, D, of SNS. In either case, the resulting statement in the conclusion has no informational value and hence, the first two statements do not lead to any useful consequence and the syllogism is said to be invalid. The check with these two criteria on the 64 examples of syllogism is being made and will be reported below. It appears that this test should reveal any other possible cause for an indefinite or doubtful output that could be present in the syllogistic argument composed of three quantified statements. (Added later: one was found namely that the property $(\forall x) (\underline{a} \Rightarrow \underline{b}) \supset (\exists x) (\underline{a} \Rightarrow \underline{b})$ has to be used sometimes, see Section 4(a)).

(Note: Since this is the first draft and the further studies are being made as the report is being written, we will not revise any of the earlier sections according to what will be revealed in these further studies. On the other hand, the new results will be described in full, and their consequences to any of the previous sections will be mentioned and clarified as we go along.)

4. Analysis of all possible 64 combinations in categorical syllogisms

The categorical syllogism has essentially the structure of Eq (2a) with the second term being the major premise and the first term being the minor premise, and these two together leading to the conclusion on the right hand side of Eq (2a).

Considering the nature of the quantifier $\forall_1, \forall_2, \forall_3$, as well as of the SNS operators $\exists_1, \exists_2, \exists_3$, we restrict ourselves to $\forall_1, \forall_2, \forall_3$ being either \forall or \exists . In the affirmative form of the statements, e.g. "All men are mortal", we take \exists to be the SNS connective \exists (as in $\underline{a} \Rightarrow \underline{b}$) and the negational sentence, e.g. "All men are not-four-legged" will be represented by using the SNS connective $\exists \mathcal{N}$ (as in $\underline{a} \Rightarrow \neg \underline{b}$). This is in the spirit in which the syllogisms have been symbolized by Church in Ref. [3]. Thus, we do not use the states in which the SNS sentence quantified by \forall or \exists is negated. Instead, the

negated form of " \underline{a} imply \underline{b} " is taken to be " \underline{a} imply $\neg \underline{b}$ ".

We have used this notation in all the previous sections, only mentioning briefly that the negation of a categorical statement puts the negation on the right hand side of the implication.

Since the four types of categorical statements do not correspond to the canonical quantifiers $\forall, \exists, \Phi, \Lambda$, we shall use the symbolism employed in the literature for these four types of statements, namely $\underline{A}, \underline{E}, \underline{I}, \underline{O}$. They stand for the following:

$$\underline{A} : \text{All } \underline{a} \text{ is } \underline{b} \equiv (\forall x) (\underline{a} \Rightarrow \underline{b}) \quad (24a)$$

$$\underline{E} : \text{All } \underline{a} \text{ is not } \underline{b} \equiv (\forall x) (\underline{a} \Rightarrow \neg \underline{b}) \quad (24b)$$

$$\underline{I} : \text{Some } \underline{a} \text{ is } \underline{b} \equiv (\exists x) (\underline{a} \Rightarrow \underline{b}) \quad (24c)$$

$$\underline{O} : \text{Some } \underline{a} \text{ is not } \underline{b} \equiv (\exists x) (\underline{a} \Rightarrow \neg \underline{b}) \quad (24d)$$

In combining these statements in the four figures to form various types of categorical syllogisms, when the first and second statements are taken together the quantified output of the first



sentence is taken as the quantified input of the second statement. When doing this, we have to apply the connective "into" defined earlier and remember that the combination $\exists \otimes \exists$ leads to the indefinite state Δ . It is readily verified that, because of this, the combinations I, IO, OI, OO will all lead to informationless conclusions in each of the four figures. Consequently, 16 of the 64 examples are inapplicable in a practical way. In the same way, we can show that the invalidity from combining the SNS connectives occurs only when IN is fed to I or IN. This is seen from Eq (25) where the four possible products between pairs of I or IN are shown.

$$\begin{matrix} \text{I} \\ \sim \end{matrix} \begin{matrix} \text{I} \\ \sim \end{matrix} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} = \begin{matrix} \text{I} \\ \sim \end{matrix} \quad (25a)$$

$$\begin{matrix} \text{I} \\ \sim \end{matrix} (\begin{matrix} \text{I} \\ \sim \end{matrix} \text{N}) = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} = \begin{matrix} \text{I} \\ \sim \end{matrix} \text{N} \quad (25b)$$

$$(\begin{matrix} \text{I} \\ \sim \end{matrix} \text{N}) \begin{matrix} \text{I} \\ \sim \end{matrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} = \text{D} \quad (25c)$$

$$(\begin{matrix} \text{I} \\ \sim \end{matrix} \text{N}) (\begin{matrix} \text{I} \\ \sim \end{matrix} \text{N}) = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} = \text{D} \quad (25d)$$

This is equivalent to saying that two successive implications will not lead to D only if the middle term is affirmative in both of them, or negational in both.

Ineffectiveness arising from the EPL state Δ , or the SNS state D, in the conclusion is obviously a general rule for not listing the particular syllogism, because it is of no value. We shall now apply these to the 16 items in each of the four figures. The scheme of implications for each is given in Table 5 below, for ready reference, although they are mentioned in most books on logic e.g. Ref. [6].

Table 5. Four figures of Traditional Logic

(a) Sixteen possible combinations in the first figure

Since both the major premise as well as the minor premise have four possible forms A, E, I, O there are 16 combinations

Table 5: Four figures of Traditional Logic

Statement	Figure 1	Figure 2	Figure 3	Figure 4
Major premise	$m \rightarrow p$	$p \leftarrow m$	$m \rightarrow p$	$p \leftarrow m$
Minor premise	$s \rightarrow m$	$s \rightarrow m$	$m \rightarrow s$	$m \rightarrow s$
Conclusion	$s \rightarrow p$	$s \rightarrow p$	$s \rightarrow p$	$s \rightarrow p$

of these that can be considered in the first figure so as to lead to a conclusion going from s to p. On examining these, it is found that the combinations EA, EE, EI, E0, OA, OE, OI, OO are useless since the SNS output is always the doubtful state (1 1), as per Eqns (25c, d). Thus eight more become invalid, and, together with the four mentioned above, for which the quantifier state is Δ , ten syllogisms within the sixteen possible in the first figure all become of no use (two of these eight, namely OI and OO, also occur in the list of four for Δ). This gives only six remaining ones and of these four are found to be listed in books dealing with traditional categorical syllogisms, namely AA, AI, EA, EI.

We shall consider why two of the six, namely IE and OA, are invalid a little later in this subsection after considering the four moods valid in figure 1. Their Latin names are also

given in the Eqns (26 a to d) below.

AAA-1 (Barbara):

$$(\forall x) (\underline{m} \Rightarrow \underline{p}), (\forall x) (\underline{s} \Rightarrow \underline{m}) ; (\forall x) (\underline{s} \Rightarrow \underline{p}) \quad (26a)$$

EAE-1 (Celarent)

$$(\forall x) (\underline{m} \Rightarrow \neg \underline{p}), (\forall x) (\underline{s} \Rightarrow \underline{m}) ; (\forall x) (\underline{s} \Rightarrow \neg \underline{p}) \quad (26b)$$

AII-1 (Darii)

$$(\forall x) (\underline{m} \Rightarrow \underline{p}), (\exists x) (\underline{s} \Rightarrow \underline{m}) ; (\exists x) (\underline{s} \Rightarrow \underline{p}) \quad (26c)$$

EIO-1 (Ferio)

$$(\forall x) (\underline{m} \Rightarrow \neg \underline{p}), (\exists x) (\underline{s} \Rightarrow \underline{m}) ; (\exists x) (\underline{s} \Rightarrow \neg \underline{p}) \quad (26d)$$

Of these, the first two only differ in that the major premise is an assertion in (26a) and a denial in (26b). The first two (26a, b) differ from the second pair (26c, d) in that, the minor premise is $(\forall x)$ in the former and $(\exists x)$ in the latter. It will be noticed that, in the way in which the two of them are applied one "into" the other, $(\exists x)$ precedes $(\forall x)$ and we have used the result of Section 2 for this — namely $\exists \oplus \exists = \exists$.

The question may be asked whether we cannot have the corresponding equations in which a statement with $(\forall x)$ precedes one having the quantifier $(\exists x)$. Actually the two possibilities of this type, with a positive and a negative statement for the major premise, are absent in the above list of four valid syllogisms. This is so, because the output of $(\forall x) (\underline{a} \Rightarrow \underline{b})$ is $(\exists x) (\underline{b})$ and it is this that is input "into" the next statement $(\exists x) (\underline{b} \Rightarrow \underline{c})$. Hence it leads to Δ and the corresponding syllogism is valueless.

Thus, the statement $(\forall x) (\underline{a}(x) \Rightarrow \underline{b}(x))$, gives, with the input $(\forall x) (\underline{a}(x))$, only an output $(\exists x) (\underline{b}(x))$, and not $(\forall x) (\underline{b}(x))$. The best way of appreciating this is to take a simple example, namely "All men are mammals", in which men is \underline{a} and mammal is \underline{b} . In fact, if we put the input "all men" $((\forall x) (\underline{a}(x)))$ into it, we will get the corresponding mammals, but it need not exhaust the domain of all mammals. We in fact

have to write the above statement more precisely as "All men

are some mammals". Hence the output of $((\forall x) (\underline{a}(x) \Rightarrow \underline{b}(x)))$

is in the state \exists . Consequently, if the next statement,

into which this output goes, has the quantifier \exists , then we get

as input to that the "into" product $\exists \otimes \exists \equiv \Delta$, resulting in

the indefinite quantifier state for the major and minor premises

put together.

If the argument of the above para is taken into account,

the two examples AE and AO go out and we are left only with

four out of the 16 possibilities under Figure 1, namely those

listed in Eqns (26 a to d). Even in these, it is readily seen

that (26b) can be derived from (26a) by putting $\neg \underline{p}$ instead

of \underline{p} and similarly (26d) can be derived from (26c).

Consequently there are really two moods in Figure 1 that are

independent — namely AAA-1 (Barbara) and AII-1 (Darii).

(b) Discussion of the second figure and reversal of QPL statements.

The second figure has the form

$$p \rightarrow m \text{ (Major)}, \quad s \rightarrow m \text{ (Minor)}, \quad s \rightarrow p \text{ (Conclusion)}$$

In this, it will be seen that the $m \rightarrow p$ (Major) of the first figure has been converted to the form $p \rightarrow m$. Thus, there is a need to work out the consequence of reversing the direction of an implication statement in QPL and the equivalences of the forward and reverse statements. (A slight repetition may be excused, as the ideas obtained at this stage have been briefly introduced earlier in the text, to make the argument clearer).

Considering the statement "All \underline{a} are \underline{b} ", it is clear that the reversal of this is "All not- \underline{b} are not- \underline{a} ". Thus for the universal quantifier, we have the following equivalences under reversal

$$(\forall x) (\underline{a} \Rightarrow \underline{b}) \equiv (\forall x) (\neg \underline{b} \Rightarrow \neg \underline{a}) \quad (27a)$$

$$(\forall x) (\underline{a} \Rightarrow \neg \underline{b}) \equiv (\forall x) (\underline{b} \Rightarrow \neg \underline{a}) \quad (27b)$$

problem is however not so simple for the existential quantifier ($\exists x$). In this case, using the standard notation $x) (\underline{a} \& \underline{b})$ for "some \underline{a} are \underline{b} ", it is clear that the statement it is quantified is symmetric in \underline{a} and \underline{b} . This can be readily verified from a Venn diagram, which shows that "some \underline{a} are \underline{b} " so means that "some \underline{b} are \underline{a} ". At the same time, from the diagram, it can be seen that when "some \underline{a} are \underline{b} ", $\neg \underline{a}$ n correspond to either \underline{b} or $\neg \underline{b}$, so that it gives the state . Similarly starting from $\neg \underline{b}$ we arrive at the state a_D . is is a new type of relationship between \underline{a} and \underline{b} which is t met with in either classical logic (propositional calculus), SNS logic. However, these properties of $(\exists x) (\underline{a} \& \underline{b})$ ll have to be borne in mind while working out the consequences f reversed statements with the quantifier ($\exists x$), going from to \underline{a} , when it has been defined for \underline{a} to \underline{b} . In view of this, e shall use the symbology $(\exists x) (\underline{a} \Rightarrow \underline{b})$ and $(\exists x) (\underline{b} \Rightarrow \underline{a})$ o denote the forward and backward relations of this type,

which have the properties mentioned above of an implication.

In the light of this, we shall write the equivalences under reversal of a CL statement which is quantified by the existential quantifier ($\exists x$) as below.

$$(\exists x) (\underline{a} \Rightarrow \underline{b}) \equiv (\exists x) (\underline{b} \Leftarrow \underline{a}) \quad (28a)$$

$$(\exists x) (\underline{a} \Rightarrow \neg \underline{b}) \equiv (\exists x) (\neg \underline{b} \Leftarrow \underline{a}) \quad (28b)$$

(It is to be noted that for (28a, b), while a_T leads to b_T or b_F as the case may be, a_F will only lead to the state b_D , as mentioned above.) Although both (28a) and (28b) are symmetric relations, we shall use the notation $(\exists x) (\underline{a} \Rightarrow \underline{b})$, or $(\exists x) (\underline{b} \Rightarrow \underline{a})$, as the case may be, to indicate the direction in which it is applied.

In addition to these, we have the property mentioned in the last sub-section for the first figure, namely $(\forall x) (\underline{a} \Rightarrow \underline{b})$, followed by $(\exists x) (\underline{b} \Rightarrow \underline{c})$, gives only $(\exists x) (c_D)$, and no useful QPL relation from $\underline{a} \rightarrowtail \underline{c}$ follows.

In other words, two categorical statements, successively having the quantifiers $(\forall x)$ and $(\exists x)$ will lead to the quantifier state Δ , even if the SNS output of the first (say \underline{b}) is the same as the input into the second statement, and no D state occurs.

Using the above results for the reversed statement and conditions of validity, we find that, in the second figure, most of the syllogisms lead either to the state Δ for the quantifier, or D for the SNS term, and are of no value. Only four of them remain, and we shall mention these below.

While working these out one more rule had to be used. Suppose, we use $(\forall x) (\underline{b} \Rightarrow \underline{a})$ in the form of $(\forall x) (\neg \underline{a} \Rightarrow \neg \underline{b})$ and combine it with another statement, which has the form $(\forall x) (\underline{b} \Rightarrow \underline{c})$, with the SNS term \underline{b} as input, then because the term put in is $\neg \underline{b}$ and the implication is from \underline{b} , we will obtain a net SNS term c_D in the D state. However, under the same conditions, we can use the fact that

$$(\forall x) (\underline{b} \Rightarrow \underline{a}) \supset (\exists x) (\underline{b} \Rightarrow \underline{a}), \quad (29)$$

which, on reversal, gives $(\exists x) (\underline{a} \Rightarrow \underline{b})$. The output $(\exists x) (\underline{b})$ of this can be the input in $(\forall x) (\underline{b} \Rightarrow \underline{c})$, and therefore, there can be a useful syllogism arising from the partial information contained by $(\exists x) (\underline{b} \Rightarrow \underline{a})$ in the form of $(\exists x) (\underline{b} \Rightarrow \underline{a})$. We will give examples of these as we consider the second, third and fourth figures.

(c) Details of the applications for the second figure.

Considering the pair of statements $p \rightarrow m$ and $s \rightarrow m$ in this second figure, we can reverse the direction of the major statement $p \rightarrow m$ to get one from $m \rightarrow p$, and also reverse the order of the two statements to get an implication from s to p .

Taking the first combination AA, this yields (30a, b):

$$(\forall x) (\underline{p} \Rightarrow \underline{m}), (\forall x) (\underline{s} \Rightarrow \underline{m}) \quad (30a)$$

$$\equiv (\forall x) (\underline{s} \Rightarrow \underline{m}), (\forall x) (\neg \underline{m} \Rightarrow \neg \underline{p}) \rightarrow \underline{p} = D \quad (30b)$$

yielding no useful conclusion. However, we can use the reversal of the quantifier $(\exists x) (\underline{p} \Rightarrow \underline{m})$ contained in the first QPL term of (30a), to obtain $(\forall x) (\underline{p} \Rightarrow \underline{m}) \supset (\exists x) (\underline{p} \Rightarrow \underline{m}) \equiv (\exists x) (\underline{m} \Rightarrow \underline{p})$. However, on putting into it the output of the statement $(\forall x) (\underline{s} \Rightarrow \underline{m})$, namely $(\exists x) (\underline{m})$, we have the case of its output $(\exists x)$ going "into" another $(\exists x)$, which, as we have already seen in Section 3, will give only the result $(\Delta x) (\underline{p})$ as the final output. Hence, in both ways of working out this combination AA in the second figure, we get either the

SNS state D or the QPL state Δ , both of which are

indeterminate. Hence AA is of no use in figure 2.

The combination AE, on the other hand, gives on reversal of $(\forall x) (\underline{s} \Rightarrow \underline{m})$ what is shown (31), which gives a useful syllogism AEE-2. This is the mood Camestres.

$$\begin{aligned}
 & (\forall x) (\underline{s} \Rightarrow \neg \underline{m}), (\forall x) (\neg \underline{m} \Rightarrow \neg \underline{p}) ; \\
 & (\forall x) (\underline{s} \Rightarrow \neg \underline{p}) \quad (\text{AEE-2})
 \end{aligned} \tag{31}$$

In the same way, we find that AI is of no use in figure 2, while AO gives the output O leading to the mood, AOO-2, namely Baroco.

Similarly, for $(\forall x) (\underline{p} \Rightarrow \neg \underline{m})$, if we combine it, after reversal, with statements of the form A, E, I, O of $s \mapsto m$, two of them lead to indefinite conclusions and two of them lead to useful syllogisms, namely EAE-2 (Cesare) and EIO-2 (Festino).

For the remaining eight combinations in figure 2, either

D or Δ results and no more useful syllogisms are obtained.

Thus, the four syllogisms mentioned under categorical syllogisms of traditional logic corresponding to the figure 2 are derivable by our method and we can also show that no others are possible, of the type $\underline{s} \Rightarrow \underline{p}$ or $\underline{s} \Rightarrow \neg \underline{p}$, quantified by \forall or \exists .

(d) Application to syllogisms in the third figure

In this figure, the statements are of the type $\underline{m} \rightarrow p$, $\underline{m} \rightarrow s$, and we can obtain a conclusion of the type $s \rightarrow p$, by reversing the second of these. Considering the first mood, AA, it is found that one of the statements, on reversal, has \underline{m} , or $\neg \underline{m}$, as output, while the other has $\neg \underline{m}$, or \underline{m} , respectively as input, leading to the SNS state D for the conclusion. On the other hand,

$$(\forall x) (\underline{m} \Rightarrow \underline{p}) \supset (\exists x) (\underline{m} \Rightarrow \underline{p}) \equiv (\exists x) (\underline{p} \Rightarrow \underline{m}) \quad (32a)$$

and, on combining with $(\forall x) (\underline{m} \Rightarrow \underline{s})$, we obtain the conclusion as in (32b)

$$(\exists x) (\underline{p} \Rightarrow \underline{m}), (\forall x) (\underline{m} \Rightarrow \frac{\Delta}{\underline{p}}) = (\exists x) (\underline{p} \Rightarrow \underline{s}) \equiv (\exists x) (\underline{s} \Rightarrow \underline{p}) \quad (32b)$$

in the I form, and we obtain the mood AAI-3 (Darapti).

In the same way, EA leads to a conclusion $\exists(\underline{s} \Rightarrow \neg \underline{p})$

in the form O, giving the valid mood EAO-3 (Felopton).

On the other hand, if we do the same procedure of deriving

an \exists from a \forall , and combine it suitably with another \forall we
for AE and EE
strange results.)
get the following. For AE, we have,

$$(\forall x) (\underline{m} \Rightarrow \underline{p}) \supset (\exists x) (\underline{m} \Rightarrow \underline{p}) \equiv (\exists x) (\underline{p} \Rightarrow \underline{m}) \quad (33a)$$

and, on combining with E, we obtain

$$\begin{aligned} (\exists x) (\underline{p} \Rightarrow \underline{m}), (\forall x) (\underline{m} \Rightarrow \neg \underline{s}) &= (\exists x) (\underline{p} \Rightarrow \neg \underline{s}) \\ &\equiv (\exists x) (\neg \underline{s} \Rightarrow \underline{p}) \end{aligned} \quad (33b)$$

Similarly, for EE, we have, for the second E,

$$\begin{aligned} (\forall x) (\underline{m} \Rightarrow \neg \underline{s}) \supset (\exists x) (\underline{m} \Rightarrow \neg \underline{s}) \\ &\equiv (\exists x) (\neg \underline{s} \Rightarrow \underline{m}) \end{aligned} \quad (34a)$$

On combining with the first E, we obtain

$$(\exists x) (\neg \underline{s} \Rightarrow \underline{m}), (\forall x) (\underline{m} \Rightarrow \neg \underline{p}) = (\exists x) (\neg \underline{s} \Rightarrow \neg \underline{p}) \quad (34b)$$

In (33b) and (34b), we obtain conclusions for $(\exists x)$, of

implications inside the bracket from $\neg \underline{s}$ to \underline{p} or to $\neg \underline{p}$.

Such conclusions like "Not- \underline{s} is \underline{p} " and "Not- \underline{s} is not- \underline{p} " have not

been considered in traditional logic, and these statements are

not derivable from $(\exists x) (\underline{s} \Rightarrow \underline{p})$ and $(\exists x) (\underline{s} \Rightarrow \neg \underline{p})$.

Hence, we shall designate forms like these by dashed symbols
as follows:

$$(\forall x) (\neg \underline{s} \Rightarrow \underline{p}) \equiv \underline{A}' , (\forall x) (\neg \underline{s} \Rightarrow \neg \underline{p}) \equiv \underline{E}' \quad (35a)$$

$$(\exists x) (\neg \underline{s} \Rightarrow \underline{p}) \equiv \underline{I}' , (\exists x) (\neg \underline{s} \Rightarrow \neg \underline{p}) \equiv \underline{O}' \quad (35b)$$

Then the moods of (33) and (34) are AEI'-3 and EEO'-3.

We shall list all the possibilities having conclusions of
the above types \underline{A}' , \underline{E}' , \underline{I}' , \underline{O}' in the next section. However,
we shall continue to consider the traditional ones with conclusions
 $\underline{A}, \underline{E}, \underline{I}, \underline{O}$ in the figures 3 and 4 in this section. // First we
shall consider the third figure which is the subject matter of
this subsection. Thus, AI and AO lead to valid conclusions as below:

$$(\forall x) (\underline{m} \Rightarrow \underline{p}), (\exists x) (\underline{m} \Rightarrow \underline{s}) \equiv (\exists x) (\underline{s} \Rightarrow \underline{m}), (\forall x) (\underline{m} \Rightarrow \underline{p}) \\ \equiv \exists (\underline{s} \Rightarrow \underline{p}) \quad \text{AII-3 (Datisti)} \quad (36a)$$

$$(\forall x) (\underline{m} \Rightarrow \neg \underline{p}), (\exists x) \underline{m} \Rightarrow \underline{s}) \equiv (\exists x) (\underline{s} \Rightarrow \underline{m}), (\forall x) (\underline{m} \Rightarrow \neg \underline{p}) \\ \equiv \exists (\underline{s} \Rightarrow \neg \underline{p}) \quad \text{EIO-3 (Feriso)} \quad (36b)$$

Two more are obtained from IA and OA, namely IAI-3 (Disamis) and AOO-3 (Bocardo). Thus, all six valid moods of the third figure are obtainable by using our technique.

In fact, six more come out to be valid, with implications starting from not-s as conclusions, namely in the forms I', E', I' and O'. As mentioned earlier, we shall list these in the next section. Thus, in the third figure all 12 valid possibilities of combining pairs of the forms A, E, I, O in the third figure have valid conclusions. The four that are not valid have the combinations (}, }), which lead to the state ∅ } = Δ for the quantifier and are therefore indefinite.

(e) Application to the syllogisms in the fourth figure

The fourth figure has the first two terms of the type

$p \rightarrow m$, $m \rightarrow s$, leading to the conclusion $s \rightarrow p$. The first one to be considered is AA for which the books on this subject write the mood to be of the form AAI-4 (Bamalip) [3], [6].

The form I represents the statement $(\exists x) (\underline{s} \Rightarrow \underline{p})$, and it is obtained from the partial information $(\exists x) (\underline{m} \Rightarrow \underline{s})$ contained in $(\forall x) (\underline{m} \Rightarrow \underline{s})$. However, it is possible to derive from AA in figure 4 a more powerful conclusion with the quantifier $(\forall x)$ throughout, as follows:

$$(\forall x) (\underline{p} \Rightarrow \underline{m}), (\forall x) (\underline{m} \Rightarrow \underline{s}) = (\forall x) (\underline{p} \Rightarrow \underline{s}) \\ \equiv (\forall x) (\neg \underline{s} \Rightarrow \neg \underline{p}) (\underline{E'}), \quad \underline{AAE}'-4 \quad (37)$$

Thus, not only can the forms A', E', I', O' occur as conclusions, but in this instance, such a conclusion is the proper one to be taken since it is superior in power to that listed commonly.

Thus, for all x , if "not- \underline{s} implies not- \underline{p} ", then obviously there exists "some x for which $\underline{s} \Rightarrow \underline{p}$ "; but there can also be "some x for which $\underline{s} \Rightarrow \neg \underline{p}$ ".

To illustrate this, we shall take the following example. Thus,

"for all" "In Bangalore \Rightarrow In Karnataka", and "for all"

"In Karnataka \Rightarrow In India" together imply, "for all" "In Bangalore \Rightarrow In India". The last statement is well-known to be equivalent to "for all" "Not in India \Rightarrow Not in Bangalore".

This is much more powerful than saying merely that "Some Indians are in Bangalore". Therefore we have a very strong case for

examining all the 64 possibilities of combining the four forms

with one another in the four figures for obtaining the new forms

of conclusion, namely A', E', I', O'. We have done this in the

next section and the essential results are summarised. We do not

give the detailed derivations for the fourth figure for the

traditional ~~words~~^{words}, but merely state that the four listed examples

other than Bamalip, namely AEE (Calemes), EAO (Fesapo), EIO

(Fresison), IAI (Dimatis) come out all right by our analysis

of all possibilities in the fourth figure.

5. New type of syllogisms leading to conclusions of the form $\neg \underline{s} \Rightarrow \underline{p}$ and $\neg \underline{s} \Rightarrow \neg \underline{p}$.

(a) Outline of the algebraic technique for working out valid syllogisms

We have so far considered the standard type of categorical syllogisms in traditional logic. While doing this, we have developed an algebraic technique of finding out which combinations of two QPL statements in the various figures will give rise to a conclusion of the type $(\forall x) (\underline{s} \Rightarrow \underline{p}, \text{ or } \neg \underline{p})$ and $(\exists x) (\underline{s} \Rightarrow \underline{p}, \text{ or } \neg \underline{p})$. The inputs also are QPL statements of the form quantified by $(\forall x)$ or $(\exists x)$. In this algebraic technique, we use reversals of these statements to obtain effectively a set of two statements in succession in which the output of the first statement is the input of the second statement. Consequently, the two quantified statements, each of which is an implication, can be combined together to form a single implication from the first term to the third term. We shall call this way of writing the two QPL statements in succession as "a chain of two implications"

or a "derivation". In effect, we have only two types of

derivations which are used, namely,

$$(\forall x) (\underline{a} \Rightarrow \underline{b}), (\forall x) (\underline{b} \Rightarrow \underline{c}) = (\forall x) (\underline{a} \Rightarrow \underline{c}) \quad (38a)$$

and

$$(\exists x) (\underline{a} \Rightarrow \underline{b}), (\forall x) (\underline{b} \Rightarrow \underline{c}) = (\exists x) (\underline{a} \Rightarrow \underline{c}) \quad (38b)$$

In order to convert the two statements in each modus of a figure into a chain of two such implications, we use reversals of one, or both, of the statements (as in 39a, b) and also use the fact (where needed) that the implication $(\forall x) (\underline{a} \Rightarrow \underline{b})$ implies the existential statement $(\exists x) (\underline{a} \Rightarrow \underline{b})$.

$$(\forall x) (\underline{a} \Rightarrow \underline{b}) \equiv (\forall x) (\neg \underline{b} \Rightarrow \neg \underline{a}) \quad (39a)$$

$$(\exists x) (\underline{a} \Rightarrow \underline{b}) \equiv (\exists x) (\underline{b} \Rightarrow \underline{a}) \quad (39b)$$

Further, we use the results from sections 2 and 3 that $\forall \otimes \forall = \forall$, $\forall \otimes \exists = \exists \otimes \forall = \exists$, and $\exists \otimes \exists = \Delta$, and also the SNS-type of result, that $(a \Rightarrow b), (\neg \underline{b} \Rightarrow \underline{c})$ in succession will lead

to the state D for the term \underline{c} irrespective of what \underline{a} is,
so that the resultant syllogism leads to no conclusion of
value, and becomes "invalid".

(b) Proof of uniqueness of the technique

As in the previous section 4, we have worked out various examples in which the above formulae (38) and (39), and the associated results mentioned above, were employed, and the 18 syllogisms of traditional logic were derived in this way.

The nineteenth one, namely Bamalip, was shown to be replaceable by a superior syllogism~~s~~ of the new type with an implication starting from $\neg \underline{s}$. It might be wondered whether this method of working out valid syllogisms, in which various reversals are used, and $(\exists x)$ is employed instead of $(\forall x)$ for a number of cases, cannot lead to two ambiguous results. We have examined this, and it is possible to show very simply that no ambiguity will come

in any case of a pair of statements, according to any of the four figures, and that, either a uniquely possible conclusion can be derived, or it can be shown that the pair of statements will lead to an indefinite statement having either the QFL state Δ , or the SNS state D. We shall give an outline of the proof below and it is not very difficult. However, the details of the proof are omitted and only the main steps are given.

While combining two statements having the quantifiers $(\forall x)$ and/or $(\exists x)$, we should first note that the combination of one statement with $(\exists x)$ followed by a second statement with $(\exists x)$ will not lead to any conclusion since $\exists \otimes \exists = \Delta$, the indefinite state. Hence we need consider only the combinations given below.

(i) If both statements are $(\forall_1 x)$ and $(\forall_2 x)$, then either they do not lead to any conclusion because the output of the

first statement is the negation of the input to the second statement, or there is a valid conclusion having the quantifier $(\forall x)$, in which case reversing both of them will lead once again to a conclusion which is equivalent to the earlier one. For example,

$$(\forall x) (\underline{a} \Rightarrow \underline{b}), (\forall x) (\underline{b} \Rightarrow \underline{c}) = (\forall x) (\underline{a} \Rightarrow \underline{c}) \quad (40a)$$

$$\begin{aligned} (\forall x) (\neg \underline{c} \Rightarrow \neg \underline{b}), (\forall x) (\neg \underline{b} \Rightarrow \neg \underline{a}) \\ = (\forall x) (\neg \underline{c} \Rightarrow \neg \underline{a}) \end{aligned} \quad (40b)$$

It is obvious that the right hand sides of (40a) and (40b) are equivalent and no ambiguity arises.

(ii) If we have the combination (\forall, \exists) or (\exists, \forall) , then one of these produces invalidity, because the sequence of statements are of the type (41), leading to an indefinite conclusion, having the QPL state Δ .

$$(\forall x) (\underline{a} \Rightarrow \underline{b}), (\exists x) (\underline{b} \text{ or } \neg \underline{b} \Rightarrow \underline{c}) = \Delta (\underline{a} \Rightarrow \underline{c}, \text{ or } \underline{a} \not\sim \underline{c}) \quad (41)$$

In such a case, only the other combination with $(\exists x)$ preceding $(\forall x)$ need be considered, and this has the quantifier $(\exists x)$, but

the SNS operator may be \Rightarrow (I), or D. In the former case, there is a unique valid syllogism, and in the latter case, no valid syllogism is possible for this combination.

(iii) In the case (i), although the combination $(\forall_1 x)$ and $(\forall_2 x)$ do not lead to any valid syllogism, we can obtain the state $(\exists x)$ by implication from one of them, and then, we can have four possible combinations. Thus, the implication being taken from either $(\forall_1 x)$ or $(\forall_2 x)$ leads to two possibilities $(\forall_1 x), (\exists_2 x)$ and $(\exists_1 x), (\forall_2 x)$, and each pair can be taken as it is, or in the reversed sense, leading to four cases. By experience, when all the 64 examples were worked out, not a single duplication, or ambiguity, was noticed. Therefore, a proof for this has been worked out and it is quite simple to see. It is best verified by saying that if there is a valid syllogism, then on reversing it and on interchanging the $(\exists x)$ and the $(\forall x)$, no other valid result comes out. Thus, suppose

$$(\exists x) (\underline{a} \Rightarrow \underline{b}), (\forall x) (\underline{b} \Rightarrow \underline{c}) = (\exists x) (\underline{a} \Rightarrow \underline{c}) \quad (42a)$$

In this \underline{a} can be either \underline{s} or $\neg \underline{s}$, and \underline{c} can be either \underline{p} or $\neg \underline{p}$. If now, we reverse the two statements on the left hand side, we obtain

$$(\forall x) (\neg \underline{c} \Rightarrow \neg \underline{b}), (\exists x) (\underline{b} \Rightarrow \underline{a}) = \forall \oplus \exists = \Delta \text{ and } D \quad (42b)$$

If, on the other hand, we interchange the $(\forall x)$ and $(\exists x)$ in (42a) then we obtain the QPL indefinite state Δ :

$$(\forall x) (\underline{a} \Rightarrow \underline{b}), (\exists x) (\underline{b} \Rightarrow \underline{c}) = \forall \oplus \exists = (\Delta x) (\underline{a} \Rightarrow \underline{c}) \quad (42c)$$

On reversing this, we obtain,

$$(\exists x) (\underline{c} \Rightarrow \underline{b}), (\forall x) (\neg \underline{b} \Rightarrow \neg \underline{a}) = (\exists x) (\underline{c} \sqcup \underline{a}) \quad (42d)$$

leading to the SNS doubtful state D. Thus, in the three cases (42b), (42c) and (42d), either the QPL state is Δ , or the SNS state is D, or both of them occur, and they are indefinite and useless as conclusions of a syllogism. Thus, we have shown quite generally that the procedure mentioned earlier in this section of working out a chain of two implications is a very general one and can lead to no ambiguities for a pair of QPL statements of Type 1.

(c) List of the eighteen new syllogisms

Since there can only be one conclusion for a pair of statements in any of the four figures, we can omit the 18 examples which are well-known in traditional logic from consideration for the purpose of deriving the set of new syllogisms. We can also similarly omit 16 examples which have the sequence (3,3) because they lead to the indefinite state Δ , by combination via the operation "into". Thus, we are left with 30 examples and it is interesting to note that 18 of them come out to be valid syllogisms of the new type in which the implication within the bracket is of the type $\neg \underline{s} \Rightarrow \underline{p}$ or $\neg \underline{s} \Rightarrow \neg \underline{p}$. We have listed these, along with the 18 syllogisms of the old type, in Table 6.

Table 6. Summary of categorical syllogisms in traditional logic, including the new possibilities

Table 7 gives the names of the valid moods of the old and new types. The first three vowels in the names give the symbols, A, E, I or O of the three statements in the syllogisms. The presence of a fourth vowel signifies that it belongs to the new type.

Table 7. Symbols and names of valid moods of the old and new types.

To distinguish the new ones from the old ones, the letter symbolism

Table 6

Summary of Categorical Syllogisms in Traditional Logic,
including the new Possibilities*

Figure+	Moods with conclusions <u>s</u> \Rightarrow <u>p</u> and <u>s</u> \Rightarrow <u>\neg p</u>	Moods with conclusions <u>\neg s</u> \Rightarrow <u>p</u> and <u>\neg s</u> \Rightarrow <u>\neg p</u>
First	<u>AAA</u> , <u>AII</u> , <u>EAE</u> , <u>EIO</u>	<u>AEI'</u> , <u>EEO'</u> , <u>IEI'</u> , <u>CEO'</u>
Second	<u>AEE</u> , <u>EAE</u> , <u>AOO</u> , <u>EIO</u>	<u>AAO'</u> , <u>EEO'</u> , <u>IEI'</u> , <u>OAI'</u>
Third	<u>AAI</u> , <u>EAQ</u> , <u>AII</u> , <u>EIO</u> , <u>IAI</u> , <u>OAO</u>	<u>AEI'</u> , <u>EEO'</u> , <u>AOI'</u> , <u>EOO'</u> , <u>IEI'</u> , <u>CEO'</u>
Fourth	<u>AEE</u> , <u>EAQ</u> , <u>EIO</u> , <u>IAI</u>	<u>AAE'</u> , <u>EEO'</u> , <u>EOO'</u> , <u>IEI'</u>

* Forms: A = All a is b, E = All a is not-b,
I = Some a is b, O = Some a is not b;
A' = All not-a is b, E' = All not-a is not-b,
I' = Some not-a is b, O' = Some not-a is not-b.

⁺ Figures: 1 : m \rightarrow p, s \rightarrow m, s \rightarrow p
2 : p \rightarrow m, s \rightarrow m, s \rightarrow p
3 : m \rightarrow p, m \rightarrow s, s \rightarrow p
4 : p \rightarrow m, m \rightarrow s, s \rightarrow p

Table 7: Symbols and Names of Valid Moods of the
Old and New Types

Old Type	New Type
<u>AAA-1</u> , Barbara	<u>AEI'-1</u> , Kaveriar
<u>AII-1</u> , Darii	<u>EEO'-1</u> , Electronic
<u>EAE-1</u> , Celarent	<u>IEI'-1</u> , Dimension
<u>EIO-1</u> , Ferio	<u>OEO'-1</u> , Ponderous
<u>AEE-2</u> , Camestres	<u>AAO'-1</u> , Bangalore
<u>AOO-2</u> , Baroco	<u>EEO'-1</u> , Neelotpal
<u>EAE-2</u> , Cesare	<u>IEI'-1</u> , Nivedita
<u>EIO-2</u> , Festino	<u>OAI'-1</u> , Moravia
<u>AAI-3</u> , Darapti	<u>AEI'-1</u> , Magnesia
<u>AII-3</u> , Datisti	<u>AOI'-1</u> , Sarojini
<u>EAO-3</u> , Felapton	<u>EEO'-1</u> , Development
<u>EIO-3</u> , Feriso	<u>EOO'-1</u> , Mesozoic
<u>IAI-3</u> , Disamis	<u>IEI'-1</u> , Indefinite
<u>OAO-3</u> , Bocardo	<u>CEO'-1</u> , Lodestone
<u>AEE-4</u> , Calemes	<u>AAE -1</u> , Management
<u>EAO-4</u> , Fesapo	<u>EEO -1</u> , Generous
<u>EIO-4</u> , Fresison	<u>EOO -1</u> , Leonora
<u>IAI-4</u> , Dimatis	<u>IEI -1</u> , Silesia

the conclusion is given an extra prime as in A', E', I' and O'.

We shall not give the derivations of all of these examples, but take only a few of them to indicate how the rules in (39a and b) and the associated results are applied for this purpose. It will be noticed that, while doing this, we are applying no new result, or technique, that has not been used in deriving the well-known 18 categorical syllogisms of traditional logic. The same rules are applied, with only the extra provision that a conclusion starting from $\neg \underline{s}$ can also be listed. It is obvious, on examining the Table 6, that the new syllogisms are not duplications, or straightforward equivalences, of the old ones. In fact, we use only statements starting from an affirmation in the first two statements on the left hand side of the syllogisms and only 64 of these are used. However, for the concluding statement we now allow the SNS statements $\neg \underline{s} \Rightarrow \underline{p}$ and $\neg \underline{s} \Rightarrow \neg \underline{p}$, starting from $\neg \underline{s}$ also, in addition to the two implications starting from \underline{s} which are listed in the left half of Table 6. It can be shown that the case of negations for the first two statements are derivable in a straightforward manner from the 36 syllogisms listed in Table 6. (See Appendix 1 for a short proof of this).

(d) New syllogisms belonging to the first figure

In the first figure, the very first combination AA leads to the well-known syllogism AAA-1 (Barbara) of the old type. Taking the next one AE which is not in the traditional list, it consists of $(\forall x) (\underline{m} \Rightarrow \underline{p})$, $(\forall x) (\underline{s} \Rightarrow \neg \underline{m})$. From this, we obtain

$$(\forall x) (\underline{m} \Rightarrow \underline{p}) \supset (\exists x) (\underline{m} \Rightarrow \underline{p}) \equiv (\exists x) (\underline{p} \Rightarrow \underline{m}) \quad (43a)$$

$$(\forall x) (\underline{s} \Rightarrow \neg \underline{m}) \equiv (\forall x) (\underline{m} \Rightarrow \neg \underline{s}) \quad (43b)$$

$$\begin{aligned} (\exists x) (\underline{p} \Rightarrow \underline{m}), (\forall x) (\underline{m} \Rightarrow \neg \underline{s}) &= (\exists x) (\underline{p} \Rightarrow \neg \underline{s}) \\ \equiv (\exists x) (\neg \underline{s} \Rightarrow \underline{p}) - I' \quad (\text{AEI}'-1) \end{aligned} \quad (43c)$$

The conclusion is of the form I' , leading to the form AEI'-1,

for which we have given the name Kaveriar. This is based on the name (Kaveri) of one of the important rivers of Karnataka state in which Bangalore is located. The termination 'ar' means 'river' in Tamil.

The next combination AI leads to AII-1, which is Darii of

the traditional list. The one after this, namely A0, leads to no useful conclusion, but only to either D or Δ , as in (44a) and (44b). Thus,

$$(\forall x)(\underline{m} \Rightarrow \underline{p}), (\exists x)(\underline{s} \Rightarrow \neg \underline{m}) \text{ leads to } D \quad (44a)$$

Reversing both, we obtain the following inputs, leading to a Δ .

$$(\forall x)(\neg \underline{p} \Rightarrow \neg \underline{m}), (\exists x)(\neg \underline{m} \Rightarrow \underline{s}) = (\Delta x)(\neg \underline{p} \Rightarrow \underline{s}) \quad (44b)$$

Just like AI, the combination EE also leads only to an $(\exists x)$ conclusion, producing EEO'-1, and, just like A0, E0 also leads only to D or Δ . Similarly IA and OA lead to only D or Δ , while IE and OE lead to the valid syllogisms of the new type IEI'-1 and OEO'-1.

As already mentioned, the quantifier combinations in II, I0, OI and OO all lead to $\exists \oplus \exists = \Delta$, and they cannot lead to valid syllogisms. Hence, just as with the old examples, the first figure contains only four of the new type of syllogism.

(e) New syllogisms in the second, third and fourth figures

Table 6 lists these new results. Since their derivation, from a pair of QPL statements, converted to form a chain of two implications, with the sequence \forall, \forall or \exists, \forall , does not involve any technique not already described and used for the traditional examples, we do not work them out in detail.

6. Manipulations with Statements of Type 1 in QPL

Although the 64 binary combinations of the statements involved in categorical syllogisms are all capable of being represented by another statement of this type, and these have been listed in the previous section, it would be worthwhile mentioning the general rules employed in working out the outputs of statements of this type, for all possible inputs. Suppose we have the QPL term $(\forall x) (\underline{a} \Rightarrow \underline{b})$ or $(\exists x) (\underline{a} \Rightarrow \underline{b})$, then its outputs, corresponding to the inputs $(\forall x) (\underline{a}, \text{or } \neg \underline{a})$ and

($\exists x$) (\underline{a}) or ($\neg \underline{a}$), are as given in Table 8. (The proofs of these are implicit in the discussion of previous sections.)

Table 8. Algorithms for obtaining the outputs of Type 1 statements of QPL

In compiling Table 8, formulae in SNS, and the "into" connective in QPL, described in this report, have been employed. The very small number of three valid deductions (Sl.Nos 1, 3 and 5 of Table 8) can be readily computerized. While testing this procedure for routine application for a pair of statements, it was found that it is necessary to have also the "implicate" operator (\Leftarrow) in addition to the connective operator "imply" (\Rightarrow). Consequently, the three cases where this connective \exists can give an output that is not D or Δ are also included in Table 8. (Their full capabilities and application have not been worked out.)

However, initial trials show that a ~~series of~~ ^{set of two} statements having

Table 8. Algorithms for obtaining the outputs
of Type 1 statements of QPL

Sl. No.	Input	Statement	Output
1.	$(\forall x) (\underline{a}(x))$	<u>Implies (\Rightarrow)</u> $(\forall x) (\underline{a} \Rightarrow \underline{b})$	$(\exists x) (\underline{b}(x))$
2.	$(\forall x) (\neg \underline{a}(x))$	$(\forall x) (\underline{a} \Rightarrow \underline{b})$	D
3.	$(\exists x) (\underline{a}(x))$	$(\forall x) (\underline{a} \Rightarrow \underline{b})$	$(\exists x) (\underline{b}(x))$
4.	$(\exists x) (\neg \underline{a}(x))$	$(\forall x) (\underline{a} \Rightarrow \underline{b})$	D
5.	$(\forall x) (\underline{a}(x))$	$(\exists x) (\underline{a}(x) \Rightarrow \underline{b}(x))$	$(\exists x) (\underline{b}(x))$
6.	$(\forall x) (\neg \underline{a}(x))$	$(\exists x) (\underline{a}(x) \Rightarrow \underline{b}(x))$	D
7.	$(\exists x) (\underline{a}(x))$	$(\exists x) (\underline{a}(x) \Rightarrow \underline{b}(x))$	Δ
8.	$(\exists x) (\neg \underline{a}(x))$	$(\exists x) (\underline{a}(x) \Rightarrow \underline{b}(x))$	Δ and D
		<u>Implicates (\Leftarrow)</u>	
9.	$(\forall x) (\underline{a}(x))$	$(\forall x) (\underline{b}(x) \Leftarrow \underline{a}(x))$	$(\exists x) (\underline{b}(x))$
10.	$(\exists x) (\underline{a}(x))$	$(\forall x) (\underline{b}(x) \Leftarrow \underline{a}(x))$	$(\exists x) (\underline{b}(x))$
11.	$(\forall x) (\underline{a}(x))$	$(\exists x) (\underline{b}(x) \Leftarrow \underline{a}(x))$	$(\exists x) (\underline{b}(x))$

Rules for reversal

$$(\forall x) (\underline{a} \Rightarrow \underline{b}) \equiv (\forall x) (\neg \underline{b} \Rightarrow \neg \underline{a}); (\exists x) (\underline{a} \Rightarrow \underline{b}) \equiv (\exists x) (\underline{b} \Rightarrow \underline{a})$$

$$(\forall x) (\underline{b} \Leftarrow \underline{a}) \equiv (\forall x) (\neg \underline{a} \Leftarrow \neg \underline{b}); (\exists x) (\underline{b} \Leftarrow \underline{a}) \equiv (\exists x) (\underline{a} \Leftarrow \underline{b})$$

implications produce some difficulties in implementing them in the sequence in which they are written.

We shall illustrate this below by one example of the pair of statements taken from one of the 64 moods of categorical syllogisms, namely OAI'-2. These are $(\exists x) (p \Rightarrow \neg \underline{m})$, $(\forall x) (\underline{s} \Rightarrow \underline{m})$. We wish to find out the output state of \underline{p} , when the input states of \underline{s} are T and F. Thus the input $(\forall x) (s_T)$ put into $(\forall x) (\underline{s} \Rightarrow \underline{m})$, gives $(\exists x) (m_T)$. This, going into the reversed first term, namely $(\exists x) (\neg \underline{m} \Rightarrow \underline{p})$ gives $(\Delta x) (p_D)$, i.e a Δ quantifier and a D state. Now, if we input $(\forall x) (s_F)$ into $(\forall x) (\underline{s} \Rightarrow \underline{m})$, the output is m_D and this, going into $(\exists x) (\neg \underline{m} \Rightarrow \underline{p})$, gives again p_D . Thus, we get the wrong result that both $(\forall x) (s_T)$ and $(\forall x) (s_F)$ leads to the output of \underline{p} in the D state, showing that the two given QPL statements, in the sequence given, lead to an invalid syllogism. But, our Table 6 gives that the two together are valid and yield the statement $(\exists x) (\neg \underline{s} \Rightarrow \underline{p})$, so that, for the input $(\forall x) (s_F)$, the output is $(\exists x) (p_T)$.

It may be mentioned that this difficulty does not arise in SIS, where the forward (\Rightarrow) and reverse (\Leftarrow) implications are equivalent, and both will give the same result, and no difficulty arises. Thus,

$$(\underline{a} \Rightarrow \underline{b}), (\underline{b} \Rightarrow \underline{c}) = (\underline{a} \Rightarrow \underline{c}) \quad (45)$$

also,

$$(\underline{a} \Rightarrow \underline{b}) \equiv (\neg \underline{b} \Leftarrow \neg \underline{a}) \quad (46a)$$

$$(\underline{b} \Rightarrow \underline{c}) \equiv (\neg \underline{c} \Leftarrow \neg \underline{b}) \quad (46b)$$

and we get

$$\begin{aligned} (\neg \underline{c} \Leftarrow \neg \underline{b}), (\neg \underline{b} \Leftarrow \neg \underline{a}) &= (\neg \underline{c} \Leftarrow \neg \underline{a}) \\ &\equiv (\underline{a} \Rightarrow \underline{c}) \end{aligned} \quad (47)$$

On the other hand, the special situation, as mentioned above,

arises in QPL for statements of Type 1, because $(\forall x) (\underline{a} \Rightarrow \underline{b})$ and $(\exists x) (\underline{a} \Rightarrow \underline{b})$, on reversal, behave differently (see Table 8).

Thus, in an argument composed of statements of Type 1 in QPL,

it appears as if the input term cannot be applied to the first statement in order to obtain its output, it is then applied ^{which} \wedge

into the second statement and its output obtained, and so on —

a process which we have developed in great detail in connection

propositional calculus, or

with the arguments in SNS logic. It looks as if a sequence of

statements will have to be manipulated as such in many situations,

and not one by one as we would imagine. This is the lesson, that

the studies made on the 64 categorical syllogisms that are

possible, has given us.

It will be apparent, from all these, that algorithmic rules

will have to be carefully formulated so that the truth value

and quantifier value of a term can be uniquely worked out for a

sequence of Type 1 statements in QPL. The full examination of

this requires extended study and is being reserved for a later

report. However, one result seems to be clear. If we have a

chain of n QPL implications, then either the resultant implication

is always Δ or/and D; or there is just one combination of these

that leads to single QPL statement of type 1, and all the remaining

$(2^n - 1)$ possible equivalent ways of writing the chain lead to the

net resultant having either a Δ or/and a D. Hence, given a QPL input, an argument composed of a finite number of quantified categorical statements can always be worked out to yield a unique output. These very brief comments will be clarified and extended in the next report, but an outline of the proof is given in Appendix 2.

References

1. Ramachandran, G.N. Current Science (1983) 52, 293-302
2. Ramachandran, G.N. Current Science (1983) 52, 335-347
3. Church, A. Article on "Logic" in Encyclopaedia Britannica 1966 Edn., Vol. 14, p. 216, Chicago.
4. Stolle, R.R. "Set Theory and Logic", (1962), Freeman, San Francisco
5. Suppes, P. "Introduction to Logic" (1957), Van Nostrand, New York.
6. Delong, H. "A Profile of Mathematical Logic", (1971), Addison-Wesley, Reading.

Appendix 1

General algorithm for finding the resultant of two categorical statements.

It is possible to work out complete general rules for combining two categorical statements (of Type 1) to obtain a resultant statement of the same type. The treatment here deals with any categorical statement, either of the types A, E, I, O or of the types A', E', I', O'. Thus, the statements combined

can have either an affirmation or a negation as input, and similarly either a negation or an affirmation as output. We show

that two statements with the quantifiers $(\forall x)$ or $(\exists x)$ for either of them, on combining together, lead to a third statement

of the same category and that the only formulae for combination

that we need employ are what ~~is~~ contained in the categorical

syllogisms AAA-1 (Barbara) or AII-1 (Darii).

(a) One quantifier \exists and the other \forall

We have already seen that two categorical statements, in which one has the existential quantifier $(\exists x)$ and the other the

universal quantifier ($\forall x$), can combine in two different ways, with the common middle term (\underline{m}) being the output of the first statement and the input of the second statement. However, depending upon the sequence (\exists, \forall) or (\forall, \exists), only one of them (in this ^{case,} ~~the former~~) can lead to a non- Δ quantifier. In either case, we verify that the middle term \underline{m} appears with an affirmation in both of them, or a negation in both of them, so that it can go in from the first statement to the second statement. If this happens for a resultant non- Δ quantifier, then we have the output based on AAA-I (Darii), namely

$$(\exists x) (\underline{a} \Rightarrow \underline{m}), (\forall x) (\underline{m} \Rightarrow \underline{b}) = (\exists x) (\underline{a} \Rightarrow \underline{b}) \quad (A1)$$

We shall illustrate this by a random example, $(\forall x) (\neg \underline{s} \Rightarrow \underline{m})$ $(\exists x) (\neg \underline{m} \Rightarrow \underline{p})$. In this case, the sequence in which the two are given leads to a state (Δx) for the resultant. So we reverse each of them and obtain,

$$(\exists x) (\underline{p} \Rightarrow \neg \underline{m}), (\forall x) (\neg \underline{m} \Rightarrow \underline{s}) = (\exists x) (\underline{p} \Rightarrow \underline{s}) \quad (A2)$$

For doing this, we associate \underline{p} with \underline{a} of (A1), $\neg \underline{m}$ of (A2) with \underline{m} of (A1), and \underline{s} with \underline{b} of (A1). It is readily verified that this technique can be applied to any combination of one \exists with one \forall , in the two statements that are under consideration.

One of the two possibilities only can ~~at~~most form a valid non-(Δ_x) conclusion, in which case it is unique. Both of them may also lead to indefinite conclusions, having either Δ or D , in which case the pair of statements started with has no valid combination except the indefinite state.

(b) Both quantifiers \forall

If the two statements have \forall , then, as they are, we need consider only any combination of ~~it~~^{them} in which the middle term is output ~~to~~^{of} the first and the input to the next. If the two middle terms have the same sign (affirmation or negation) then it has a valid resultant, for example,

$$(\forall_1 x) (\underline{s} \Rightarrow \neg \underline{m}), (\forall_2 x) (\neg \underline{m} \Rightarrow \underline{p}) = (\forall x) (\underline{s} \Rightarrow \underline{p}) \quad (A3)$$

For obtaining this, we use the standard syllogism AAA-1 (Barbara).

which we shall write as

$$(\forall x) (\underline{a} \Rightarrow \underline{m}), (\forall x) (\underline{m} \Rightarrow \underline{b}) = (\forall x) (\underline{a} \Rightarrow \underline{b}) \quad (A4)$$

In these, the relationship between (A3) and (A4) is that we associate \underline{s} with \underline{a} , $\neg \underline{m}$ with \underline{m} and \underline{p} with \underline{b} of Barbara (A4).

It can quite happen that the two sentences, both with the universal quantifier $(\forall x)$, lead only to the doubtful state D for the resultant, e.g.

$$(\forall_1 x) (\neg \underline{s} \Rightarrow \underline{m}), (\forall_2 x) (\neg \underline{m} \Rightarrow \neg \underline{p}) = D \quad (A5)$$

In such a case, we can look for a way of combining the two to obtain a conclusion of lesser power. We do this by converting either one of the two statements into an existential statement by implication. We have already shown, in Section 5(b), that this process can lead only to one of the four ways of combining a pair of } and \forall statements, leading to a valid conclusion, and that there is no duplication or ambiguity. Hence we only illustrate the technique below, with the example of (A5).

Converting first ($\forall_1 x$) into ($\exists x$), we obtain the pair of statements

$$(\exists x) (\neg \underline{s} \Rightarrow \underline{m}), (\forall x) (\neg \underline{m} \Rightarrow \neg \underline{p}) = D \quad (A6a)$$

Reversing the two statements on the l.h.s. of (A6a), we obtain

$$(\forall x) (\underline{p} \Rightarrow \underline{m}), (\exists x) (\underline{m} \Rightarrow \neg \underline{s}) = \Delta \quad (A6b)$$

This leads, however, to Δ . Therefore, we convert ($\forall_2 x$) alone into ($\exists x$), and obtain (A6c),

$$(\forall x) (\neg \underline{s} \Rightarrow \underline{m}), (\exists x) (\neg \underline{m} \Rightarrow \neg \underline{p}) = \Delta \text{ and } D \quad (A6c)$$

and, on reversing the two statements on the l.h.s. of this, we get a valid conclusion, as in (A6d):

$$(\exists x) (\neg \underline{p} \Rightarrow \neg \underline{m}), (\forall x) (\neg \underline{m} \Rightarrow \underline{s}) = (\exists x) (\neg \underline{p} \Rightarrow \underline{s}) \quad (A6d)$$

Thus, with (\forall_1, \forall_2) as the first two statements of a syllogism, we obtain as conclusion either a conclusion with ($\forall x$), using AAA-1, or a conclusion with ($\exists x$) as quantifier, if the former is not possible. In the former case, the conclusion is straightforward, and in the latter case, it is one of the four possibilities thus obtained, using AII-1, but is unique.

(c) Both quantifiers \exists .

In this case, since $\exists \otimes \exists = \Delta$, no valid conclusion can be obtained, and such a pair of statements leads to no definite conclusion.

(d) Reversal of the resultant

It is to be noted that conclusions such as the right hand sides of (A4) and (A6d) are not the only ones that are obtained on combining the two statements on the left hand sides of such equations. Their reversals are equally valid as conclusions.

Thus, we have also

$$(A4) \equiv (\forall x) (\neg b \Rightarrow \neg a)$$

and

$$(A6d) \equiv (\exists x) (\underline{s} \Rightarrow \neg p)$$

A case of special interest where such a reversal is made after applying Darii is the derivation of OAI'-2 (Moravia).

This syllogism reads as follows:

$$(\exists x) (\underline{p} \Rightarrow \neg \underline{m}), (\forall x) (\underline{s} \Rightarrow \underline{m}) = (\exists x) (\neg \underline{s} \Rightarrow \underline{p}) \quad (A7a)$$

.72a.

The first step is to reverse the term $(\forall x) (\underline{s} \Rightarrow \underline{m})$ to get its equivalent $(\forall x) (\neg \underline{m} \Rightarrow \neg \underline{s})$. Then we obtain the following, having the pattern of Darii, whose conclusion is given in (A7b) below.

$$(\exists x) (\underline{p} \Rightarrow \neg \underline{m}), (\forall x) (\neg \underline{m} \Rightarrow \neg \underline{s}) = (\exists x) (\underline{p} \Rightarrow \neg \underline{s}) \quad (A7)$$

On reversing the conclusion in (A7b), we obtain the right hand side of (A7a), thus proving the syllogism Moravia, using Darii.

The pattern of Moravia is of great interest in relation to the working out of the conclusion of n statements, under certain conditions (see Appendix 2).

Thus, we can test if any pair of categorical statements have a valid conclusion. The conclusion can also be derived in a straightforward manner by the procedure described here. We need, for this, only the two syllogisms AAA-1 (Barbara) and AII-1 (Darii). It is obvious that this technique can be readily converted into an algorithm and computerized, but this is not presented here.

Appendix 2

Procedure for finding the resultant of a chain of n categorical statements

Suppose that we have a set of $(n + 1)$ terms $\underline{s}, m_1, m_2, \dots, m_j, \dots, m_{n - 1}, p$ and n relations connecting them made up of statements of Type 1. Each such statement will be of one of the forms A, E, I, O, A', E', I', O', and therefore can have the quantifier \forall or \exists , and can have either affirmative or negative inputs and outputs inside the bracket that is quantified. Then, if every middle term occurs as an input in one statement and an output in a different statement, then n statements connecting the above $(n + 1)$ terms can be written in the form of a chain as follows:

$$\mathcal{B}_1 \equiv (\forall x) (\underline{s} \Rightarrow b_1) \quad (A8a)$$

$$\mathcal{B}_2 \equiv (\forall x) (a_2 \Rightarrow b_2) \quad (A8b)$$

$$\mathcal{B}_j \equiv (\forall x) (a_j \Rightarrow b_j) \quad (A8c)$$

$$\mathcal{B}_n \equiv (\forall x) (a_n \Rightarrow p) \quad (A8d)$$

The condition for the chain is that the output of \mathcal{B}_1 is the input to \mathcal{B}_2 and in general, the output of \mathcal{B}_j is the input

to β_{j+1} . We shall give below the conditions under which such a chain of implications has a valid conclusion and the formulae for the conclusions when they are valid.

(a) First quantifier is \exists and all others \forall

The quantifiers $(\forall_1 x), \dots, (\forall_n x)$ can all be only $(\forall x)$ or $(\exists x)$. We also know further that irrespective of the input to a categorical statement in the form of a universal or existential quantifier, the output is always $(\exists x)$ (see Table 8). Hence no middle QPL term in (A8) can have an $(\exists x)$, since $\exists \otimes \exists = \Delta$, and this generates the Δ state, which is then carried through to the end, so that the conclusion is also an indefinite state. On the other hand, the first statement can be an $(\exists x)$ and all the rest can be $(\forall x)$ in which case, under suitable conditions, a valid conclusion can be arrived at. For obtaining such a valid conclusion an additional set of equations

(which we may call as the "transitivity" conditions), as in (A9), are required:

$$\underline{b}_1 = \underline{a}_2, \underline{b}_2 = \underline{a}_3, \dots, \underline{b}_j = \underline{a}_{j+1}, \dots, \underline{b}_{n-1} = \underline{a}_n \quad (A9)$$

When these are satisfied, we call the chain as a "connected chain".

If the chain is not connected, and any one of the equations in

(A9) is not satisfied, and some $\underline{b}_j = \neg \underline{a}_{j+1}$, then the input to

\underline{b}_{j+1} is F, so that the SNS state of the output from the implication is D, which is then carried over right upto the end and the final output is in the doubtful state.

Suppose, we have both the conditions mentioned above satisfied, then the statements are implementable in serial order from

\underline{b}_1 to \underline{b}_n and the resultant is

$$Y = (\exists x) (\underline{s} \Rightarrow \underline{p}) \quad A10$$

(b) All quantifiers are \forall .

If all the quantifiers are \forall , and the transitivity conditions (A9) are all satisfied, then the conclusion is

$$H' = (\forall x) (s \Rightarrow p) \quad (A11)$$

Under these conditions, all the equations in (A8) are reversible

and we can write \mathcal{B}_n to \mathcal{B}_1 in serial order as follows:

$$\beta_n \equiv (\forall x) (\neg p \Rightarrow \neg a_n) \quad (A12a)$$

$$M_{n-1} \equiv (\forall x) (\neg b_{n-1} \Rightarrow \neg a_{n-1}) \quad (A12b)$$

$$B_j \equiv (\forall x) (\neg b_j \Rightarrow \neg a_j) \quad (A12c)$$

$$I_1 \equiv (\forall x) (\neg b_1 \Rightarrow \neg s) \quad (A12d)$$

In this case, the equations in (A12) can be implemented in serial

order as written, and the resultant becomes

$$\mathcal{H}^t = (\forall x) (\neg \sqsubseteq_p \Rightarrow \sqsubseteq_S) \quad (A13)$$

(A11) and (A13) are equivalent by reversal and are not two

different conclusions. Hence a set of statements of Type 1,

all with \forall , which form a connected chain, has a unique conclusion,

in which the first term in the sequence (s) implies the last term

in the sequence (p) under the universal quantifier $(\forall x)$.

(c) Last statement alone has \exists and all others \forall .

If this happens, we reverse all the QPL sentences in (A8) and we obtain a sequence of the same type as (A12), but in which the first statement (A12a) alone becomes (A14), and all the others are unaltered.

$$\mathcal{B}_n \equiv (\exists x) (\underline{p} \Rightarrow \underline{a}_n) \quad (A14)$$

Therefore, we need new set of transitivity conditions (A15) which is the same as (A9), except that the n th one has a negation in addition, as in (A15) below.

$$\underline{a}_n = \neg \underline{b}_n, \underline{a}_{n-1} = \underline{b}_{n-1}, \dots, \underline{a}_1 = \underline{b}_1 \quad (A15)$$

They are listed in the reverse sequence from n to 1 in (A15), because they are applied that way. Then the set of statements \mathcal{B}_n to \mathcal{B}_1 can be implemented in the reverse order and we get the resultant,

$$\mathcal{X}''' = (\exists x) (\underline{p} \Rightarrow \neg \underline{s}) \equiv (\exists x) (\neg \underline{s} \Rightarrow \underline{p}) \quad (A16)$$

(d) Summary of this appendix

Summarising what we have said above, a set of n categorical statements have a valid conclusion only under the following three conditions:

1) The conclusion is $(\forall x) (\underline{s} \Rightarrow \underline{p})$ if all the quantifiers are $(\forall x)$ and the sequence of statements form a connected chain by satisfying (A9). This is the extension of AAA-1 (Barbara), from 2 to n statements, by adding any number of them, with $(\forall x)$, to the right; and the conclusion is the same as in Barbara.

2) The conclusion is $(\exists x) (\underline{s} \Rightarrow \underline{p})$ if the first statement is $(\exists x)$ and all the rest are $(\forall x)$, and they form a connected chain satisfying (A9). This is obtained by extending AII-1 (Darii), as above to n statements by adding any number of them with $(\forall x)$ to the right, and the conclusion is the same as in Darii.

3) The conclusion is $(\exists x) (\neg \underline{s} \Rightarrow \underline{p})$, if the last statement is an $(\exists x)$ and all the preceding ones are $(\forall x)$ and the connected

chain is formed in the reverse sense from the n th statement to the first one via a sequence of implications, and the transitivity equations in (A15) are satisfied. Effectively, this is also an extension of Darii, but using reverse implications, (\Leftarrow), whose properties are indicated in Table 8. It can also be considered as an extension of OAI'-2 (Moravia), from 2 to n statements, by adding any number of them with ($\forall x$), but to the left. However, the validity of Moravia itself rests on that of Darii, so that we need only Barbara and Darii, for proving the validity of any chain of Type 1 statements.

It appears as if no valid conclusion is possible under any other conditions, namely (i) if the chain is not connected by the transitivity conditions, or (ii) if there is an ($\exists x$) in any statement other than in the first, or the last, position in the chain.

The discussion in this appendix has not been thoroughly rechecked, and it is given here to provoke discussion, rather than as definitely conclusive results, as they appear to be novel.

Vector-Matrix Representation of Boolean Algebras and Application
to Extended Predicate Logic (EPL)

Part III - Algebraic Theory for Categorical Statements in
Quantified Predicate Logic (QPL)*

G.N. Ramachandran
Gita, 5, 10A Main Road
Malleswaram West
Bangalore 560 055

*Matphil Reports No. 34, December 1983

Vector-Matrix Representation of Boolean Algebras and Application
to Extended Predicate Logic (EPL)

Part III — Algebraic Theory for Categorical Statements in
Quantified Predicate Logic (QPL)*

G.N. Ramachandran
Gita, 5, 10A Main Road
Malleeswaram West
Bangalore 560055

*Matphil Reports No. 34, December 1983

Abstract

The paper deals with the application of QPL to categorical syllogisms of traditional logic. It is pointed out that the so-called "doubtful" state (D) of SNS logic and the "indefinite" state (Δ) of extended predicate logic occur naturally in the manipulation of QPL sentences having inputs with the quantifiers \forall and \exists . When these are taken into account, and the reversal equivalences $(\forall x) (\underline{a} \Rightarrow \underline{b}) \equiv (\forall x) (\neg \underline{b} \Rightarrow \neg \underline{a})$ and $(\exists x) (\underline{a} \Rightarrow \underline{b}) \equiv (\exists x) (\underline{b} \Rightarrow \underline{a})$ are also taken into account, along with the inclusion condition $(\forall x) (\underline{a} \Rightarrow \underline{b}) \supset (\exists x) (\underline{a} \Rightarrow \underline{b})$, all the 18 categorical syllogisms found in standard treatments can be algebraically derived, by reversing either, or both, of the statements in the two input premises, and checking all the four possible rearrangements for a definite conclusion. It is proved that at most one of the four leads to a definite conclusion and the ^{other} three lead either to Δ for the quantifier or D for the state of the SNS relation that is quantified. These considerations

.ii.

are summarized in a set of four algorithmic procedures, using

which the validity or non-validity of any mood out of the 64

that are possible (by using the standard forms A, E, I, O with

$\underline{s} \Rightarrow p$ or $\underline{s} \Rightarrow \neg \underline{p}$) in the four figures can be established

and the conclusion derived (as one of the forms A, E, I, O)

when it is valid. Surprisingly, not only was it possible to

derive the 18 syllogisms listed traditionally, but in addition,

18 more valid new ones could be derived which, however, have

conclusions in one of the four forms A', E', I', O' (newly defined,

having $\neg \underline{s} \Rightarrow p$ and $\neg \underline{s} \Rightarrow \neg \underline{p}$ inside the bracket that is

quantified). These have been listed and mnemonic names coined

for them. A very general theorem is obtained, namely that any

pair of categorical statements, with \forall or \exists as quantifier, has

no valid conclusion if both are \exists, \exists ; has always a valid

conclusion if both are \forall, \forall ; but may be valid or invalid if one

has the quantifier \forall and the other \exists . In proving this, it is

shown that every pair of categorical statements can be brought

.iii.

into isomorphism with the pair in one of the two classical syllogisms Barbara or Darii, which are the only independent patterns (moods) possible for the pair. This has also been extended to the case of n categorical statements in sequence, and it is shown that there are only three independent patterns possible for them to have a valid conclusion — one as an extension of Barbara with all statements quantified by \forall , and two as extensions of Darii, with one } and all the rest being \forall .

CONTENTS

	<u>Page No.</u>
1. Introduction	1
2. Outline of SNS logic with special reference to the connective for implication.	
(a) Description of the states and the unary connective "implies"	8
(b) Two implications in succession in PC	14
(c) A number (n) of implications in sequence	16
3. Essentials of QPL needed for our derivations	
(a) Sentences and states in EPL	18
(b) Outputs of single implication statements of Type-1 in QPL	19
(c) Reversal of quantified implications	20
(d) Conclusions from two categorical statements	21
4. Theory of categorical syllogisms and the newly added examples.	
(a) General principles	28
(b) Table of categorical syllogisms	41
(c) Illustrations of the procedures (i) to (iv) of Section 4(a)	44
(d) Proof of uniqueness of the procedures (i) to (iv) for obtaining the resultant	49
(e) Summary of Section 4	53
5. Resultant of n connected categorical statements	
(a) Three possible patterns leading to conclusions	55
(b) Analogue of <u>Barbara</u>	56
(c) Two analogues of <u>Darii</u> in the forward and reverse directions.	56

Part III — Algebraic Theory for Categorical Statements
in Quantified Predicate Logic (QFL)

1. Introduction

Studies on the representation of logical connectives by operators based on Boolean vectors and matrices were started some five years ago with special reference to the computerization of logic. The earlier studies were written up in the form of reports from the Mathematical Philosophy Group of the Indian Institute of Science (Ramachandran, 1979, 1980; Ramachandran and Thanaraj, 1980a, 1980b, 1981; H. Ramachandran and Thanaraj, 1981). From 1981 onwards, a very general Boolean vector-matrix algebra was developed for sentential logic which extended the standard propositional calculus with two truth values T and F, to a formalism isomorphic with the Boolean algebra/^{of} genus 2 having four states — which in logic are T(true), F(false), D(doubtful) and X(impossible). An electronic analog machine was also constructed (Ramachandran, Johnson and Thanaraj, 1980), using standard logic

.2.

gates to represent all the operators of this extended propositional

calculus, for which the name Syād-Nyāya-System logic was given

(syād = may be, nyāya = logic, in Sanskrit). The electronic

circuits in this machine closely represent the algebraic formalism

that was developed for this purpose. A brief summary of all these

was written up as a paper for Current Science (Ramachandran, 1982)

and it summarized the properties of the Boolean algebraic

representation of both the logical states and the logical connectives

of propositional calculus, in its extended form of SNS logic.

Thereafter, the Boolean algebraic technique was extended to

predicate logic, and two papers (Part I and Part II with the same

title as the present paper) were published also in Current Science

(Ramachandran 1983 a,b).

In the papers dealing with predicate logic, it was shown that

the Boolean algebra of genus 3 and, in particular, its vector-matrix

representation by three-element Boolean vectors and 3X3 Boolean

matrices, leads to the extension of the four well-known states in quantified predicate logic (QPL), (namely "for all", "there exists", "not for all" and "for none"), by adding four more states ("for some only", "for all or none", "indefinite" and "impossible"). Also the nature of the QPL connectives between such quantified SNS terms, as in $(\forall x) (\underline{a}(x)) \Rightarrow (\exists y) (\underline{b}(y))$ or $(\forall x) (\underline{a}(x)) \& (\exists x) (\underline{b}(y)) = \underline{c}$ was discussed in great detail in Ramachandran (1983 a,b). (We have given the name "Extended Predicate Logic" (EPL) for this extended form of QPL).

It was shown that logical connectives of EPL are, in general, representable by 3X3 matrices, or operators containing Boolean sums (\oplus) or products (\otimes). In fact, there are $(8 \times 8) = 64$ possible matrices having the nature of an "and" and 64 having the nature of an "or". We shall not go into the details of these here except to point out that they all deal with QPL sentences in which the connective is also a QPL operator. We shall call such sentences as "Sentences of Type-2". The formalism needed to deal with single sentences of Type-2 of all types was developed in

Parts I and II of this series and, on examining the literature, it was found that this type of connective has not been stated and investigated so extensively before.

However, it was brought to our attention that these operators, or connectives, do not deal with sentences having logical connectives of another type in QPL — such as, "Given $(\forall x) (\underline{a}(x) \Rightarrow \underline{b}(x))$ and the QPL input $(\exists x) (a_T)$, what is the QPL output containing $\underline{b}(x)?$ " As we will see later, it is $(\exists x) (b_T)$. Sentences of this type, in which the same variable x occurs throughout the sentence and both the input and the output have the same scope for its domain, may be called as "Sentences of Type-1". The purpose of this paper is to examine and discuss the properties of such sentences, pairs of them and a connected chain of them and to give the conclusion in each case for any given input, or inputs. In particular, we confine ourselves to the so-called "categorical" statements which occur in the classical syllogisms of traditional logic (Church, 1966). We shall discuss

in extenso; but we are able to design a very general procedure whereby the conclusion of any pair or statements from out of the eight forms A, E, I, O, A', E', I', O' can be obtained by using only two standard results — namely those which correspond to the classical syllogisms Barbara and Darii. Thus, it is not necessary to remember the symbols of all valid syllogism, but instead work out the conclusion of any pair of categorical statements by a straightforward derivational process by converting them into a form which is isomorphous with Barbara or Darii.

This treatment has been extended to n categorical statements in the form of a connected sequence (see Section 4 for the definition of connectedness), and it is found that, by applying the same two standard syllogisms Barbara and Darii, the conclusion, when it is valid, of an n-statement chain can be worked out, and that this occurs only for three types of sequences, whose nature can be specified. This is described in Section 5.

2. Outline of SNS logic with special reference to the connective for implication.

(a) Description of the states and the unary connective "implies"

As mentioned above, the main purpose of this paper is to derive the conditions governing a pair of (and sequences of n) categorical statements, for them to lead to a valid conclusion, and to work out formulae for deriving the conclusion in such a case. We found that the principles based on the Boolean algebraic formalism that were developed for SNS and EPL, are required for doing these. A long report (Ramachandran 1983 c) on these studies has been prepared for private circulation, and the present paper is a condensation of that, written such that only the notation and ideas commonly used in propositional calculus (PC) and QPL are employed for the proofs. Therefore, the essential ideas connected with SNS and EPL are briefly presented in this section and the next section — particularly with special reference to those aspects that find application in the theory of this paper. In particular, the Boolean algebraic representation of the logical

connectives lead to some new features regarding "implication" in Propositional Calculus and Predicate Logic. These new ideas play an important part in the derivations worked out here and their basic principles are described in Sections 2 and 3.

Table 1. Truth Table of "a implies b" and "a and b".

We shall start with the truth table of $\underline{a} \Rightarrow \underline{b}$, which is shown in Table 1(a), in which $T \mapsto 1$ (\mapsto = "corresponding to", or "leads to") and $F \mapsto 0$. The nature of the truth value of \underline{b} when $\underline{a} = F$ leads to some very interesting new ideas. Thus from Table 1a, we have

$$a_T \mapsto b_T, \text{ and } a_F \mapsto b_T \text{ or } b_F \quad (\underline{b} \text{ "may be" T or F}) \quad (3)$$

Note the slight change in nomenclature about the repercussions of \underline{a} being false, namely that \underline{b} "may be" T or F. In standard literature on logic, it is said that a_F implies both b_T and b_F . What we mean is essentially the same idea, namely that the false

Table 1. Truth Table of "a implies b" and "a and b"

		(a) <u>a</u> \Rightarrow <u>b</u>	
		T	F
		<u>a</u>	<u>b</u>
T		1	0
F		1	1

		(b) <u>a</u> & <u>b</u>	
		T	F
		<u>a</u>	<u>b</u>
T		1	0
F		0	0

truth value for \underline{a} cannot distinguish between b_T and b_F and cannot give a unique T or F truth value as the answer; but the situation is better understood by saying that, when \underline{a} is false, \underline{b} could have the truth value T, or it could have the truth value F, and there is no way of saying which is correct. In other words, the information content of what the implication $\underline{a} \Rightarrow \underline{b}$ gives, when \underline{a} is F, is nil. Thereafter, \underline{b} cannot be the input of another logical statement, connecting it, say, to \underline{c} — to give useful results about \underline{c} .

The situation when " $\underline{a} \& \underline{b}$ " is true, and \underline{a} is given, can be deduced from the truth Table 1(b). In this case also,

(4)

$a_T \mapsto b_T$, but $a_F \mapsto \neg b_T$ and $\neg b_F$ (i.e. \underline{b} can neither be true, nor false)

In this case, when \underline{a} is false, the statements " $\underline{a} \& \underline{b}$ " itself becomes meaningless for \underline{a} itself must be true for " $\underline{a} \& \underline{b}$ " to be true. In such a situation, to ask for the state of \underline{b} is

.11.

meaningless and it is an impossibility for \underline{b} to be conceived of, if \underline{a} is F, and $\underline{a} \& \underline{b}$ is true. We denote this by the symbol X standing for the "impossible" state.

A very simple representation of the "doubtful" state D and the "impossible" state X can be given by using two-element Boolean vectors $(a_\alpha \quad a_\beta)$ to represent the truth value of a term \underline{a} . We take the value 1 or 0 of a_α to indicate whether T exists or does not exist and similarly the value of a_β to indicate whether F exists or does not exist. Then, we have

$$T = (1 \quad 0), \quad F = (0 \quad 1), \quad D = (1 \quad 1), \quad X = (0 \quad 0) \quad (5)$$

The representation (1 0) for T is obvious, since truth exists and falsehood is absent, and similarly (0 1) for F. On the other hand, in the doubtful state, the term concerned can have the state T, or the state F, and we are unable to say whether it is always one or the other. Hence, the representation by (1 1)

is reasonable. On the other hand, if we take (0 0), both truth is absent and the falsehood is absent and the term is an impossible one. In general, the state X comes as the consequence of a contradiction, and though we have worked out how the consequence of a contradiction can be turned back and useful results obtained in logical arguments (as in a reductio ad absurdum proof), we shall not discuss these here, since the present paper deals essentially with implications. It may be mentioned that the output of an implication as such is never the impossible state X, but only one of T, F, or D.

We shall now focus our attention on the implication relation and its use as a unary relation between \underline{a} and \underline{b} , when $\underline{a} \Rightarrow \underline{b}$ is true. This is expressed by using a logical operator $\underline{\sim I}$ (for "implies") connecting \underline{a} and \underline{b} in the form of an equation $\underline{a} \underline{\sim I} = \underline{b}$. We have shown (see e.g. Ramachandran, 1982) that $\underline{\sim I}$ can be represented by a 2X2 Boolean matrix $|I|$, which has

for its elements the same ones as in its truth table (Table 1(a)),

and then it is readily verified that

$$(1 \ 0) \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} = (1 \ 0); \quad (0 \ 1) \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} = (1 \ 1) \quad (6a)$$

corresponding to

$$a_T \underset{\sim}{I} = b_T ; \quad a_F \underset{\sim}{I} = b_F \quad (6b)$$

The full ramifications of the Boolean vector-matrix formalism

will not be commented upon here, except ^{for} one main idea, namely

that of reversing a logical equation. Thus, if $\underline{a} \underset{\sim}{I} = \underline{b}$ is valid,

then in the reverse sense from \underline{b} to \underline{a} , there is a logical

equation. If we denote it by $\underline{b} \underset{\sim}{J} = \underline{a}$, then the matrix $|J|$ for

it is the transpose of the matrix $|I|$, for the forward relation.

This comes out of the general theory of relations (see Ramachandran,

1983a). On testing the effect of this $|J|$, we obtain, analogous

to (6), the results

$$(1 \ 0) \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} = (1 \ 1); \quad (0 \ 1) \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} = (0 \ 1) \quad (7a)$$

which yield the logical correspondences

$$b_T \underset{\sim}{\sim} a_D ; \quad b_F \underset{\sim}{\sim} a_F , \text{ or } \neg \underline{b} \Rightarrow \neg \underline{a} \quad (7b)$$

Thus, we get the consequence that an implication from \underline{a} to \underline{b}

has an equivalent logical relation from \underline{b} to \underline{a} , namely

$$(\underline{a} \Rightarrow \underline{b}) \equiv (\neg \underline{b} \Rightarrow \neg \underline{a}) \quad (7c)$$

This equivalence (or tautology) is taken as an axiom in standard

treatments of PC, but we obtain it ^{here} as a consequence of the vector-

matrix formalism, based on the definition of a logical operator

by its truth table. In this paper, we shall take the equivalence

(7) as a well established theorem and the generalization of this

to predicate calculus plays a fundamental role in our approach

to categorical syllogisms.

(b) Two implications in succession in PC

Consider the equivalence for pair of implications, usually

taken as an axiom:

$$(\underline{a} \Rightarrow \underline{b} , \underline{b} \Rightarrow \underline{c}) \equiv (\underline{a} \Rightarrow \underline{c}) \quad (8a)$$

.15.

However, this can be derived in our formulation by the fact that,

for the matrix $|I| = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$ representing the connective $\underline{\lambda}$,

the equation $|I|I| = |I|$ holds, so that

$$(\underline{a} \underline{\lambda} \underline{b} = \underline{b}, \underline{b} \underline{\lambda} \underline{c} = \underline{c}) \equiv (\underline{a} \underline{\lambda} \underline{\lambda} \equiv \underline{a} \underline{\lambda} = \underline{c}) \quad (8b)$$

However, the proof is immaterial, and we take (8) as a theorem for our treatment in this paper.

We can reverse the l.h.s. of (8) and obtain

$$(\neg \underline{c} \Rightarrow \neg \underline{b}, \neg \underline{b} \Rightarrow \neg \underline{a}) \equiv (\neg \underline{c} \Rightarrow \neg \underline{a}) \quad (9)$$

But, the right hand side of (9) is the reverse of the r.h.s. of (8a) and is equivalent to it. Hence (8a) and (9) express the same result, in two different ways, and they are equivalent to one another.

However, if in (8a) the second implication is from $\neg \underline{b}$ to \underline{c} , we obtain,

$$(\underline{a} \Rightarrow \underline{b}, \neg \underline{b} \Rightarrow \underline{c}) = \underline{a} \wedge \underline{c} = \underline{D} \quad (10)$$

The statement $\underline{a} \rightarrow (\underline{c})$ means " \underline{a} is disconnected with \underline{c} ", or both $\underline{a} = T$, and $\underline{a} = F$, put as inputs to the l.h.s. of (10), lead ^{only} to \underline{c} being D. (If $\underline{a} = T$, $\underline{b} = T$, $\neg \underline{b} = F$, leading to $\underline{c} = D$; and if $\underline{a} = F$, $\underline{b} = D$, $\neg \underline{b} = D$ and $\underline{c} = D$). Independent of our proof, (10) is an obvious result in PC. We shall use the symbol \mathbb{D} , as in r.h.s. of (10), to indicate that the l.h.s. of the equation leads to no definite consequence.

(c) A number (n) of implications in sequence

Consider

$$\underline{a}_1 \Rightarrow \underline{b}_1, \quad \underline{a}_2 \Rightarrow \underline{b}_2, \dots, \quad \underline{a}_n \Rightarrow \underline{b}_n \quad (11)$$

It is obvious that

$$\text{if } b_1 = a_2, \quad b_2 = a_3, \dots, \quad b_n = a_n \quad (12a)$$

$$\text{then } (11) \equiv (\underline{a}_1 \Rightarrow \underline{b}_n) \quad (12b)$$

On the other hand, between the ^e~~f~~quations in (11),

$$\text{if } b_j = \neg a_{j+1}, \text{ for some } j = 1, 2, \dots, n-1 \quad (13a)$$

$$\text{then } (11) \equiv (\underline{a}_1 \rightarrow \underline{b}_n) = \mathbb{D} \quad (13b)$$

.15.

However, this can be derived in our formulation by the fact that,

for the matrix $|I| = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$ representing the connective $\tilde{\wedge}$,

the equation $|I|I| = |I|$ holds, so that

$$(\underline{a} \tilde{\wedge} \underline{b} = \underline{b}, \underline{b} \tilde{\wedge} \underline{c} = \underline{c}) \equiv (\underline{a} \tilde{\wedge} \underline{b} \tilde{\wedge} \underline{c} \equiv \underline{a} \tilde{\wedge} \underline{c} = \underline{c}) \quad (8b)$$

However, the proof is immaterial, and we take (8) as a theorem for our treatment in this paper.

We can reverse the l.h.s. of (8) and obtain

$$(\neg \underline{c} \Rightarrow \neg \underline{b}, \neg \underline{b} \Rightarrow \neg \underline{a}) \equiv (\neg \underline{c} \Rightarrow \neg \underline{a}) \quad (9)$$

But, the right hand side of (9) is the reverse of the r.h.s. of (8a) and is equivalent to it. Hence (8a) and (9) express the same result, in two different ways, and they are equivalent to one another.

However, if in (8a) the second implication is from $\neg \underline{b}$ to \underline{c} , we obtain,

$$(\underline{a} \Rightarrow \underline{b}, \neg \underline{b} \Rightarrow \underline{c}) = \underline{a} \quad (\underline{c} = \underline{D}) \quad (10)$$

The statement $\underline{a} \rightarrow \underline{c}$ means " \underline{a} is disconnected with \underline{c} ", or both $\underline{a} = T$, and $\underline{a} = F$, put as inputs to the l.h.s. of (10), lead^{only} to \underline{c} being D. (If $\underline{a} = T$, $\underline{b} = T$, $\neg \underline{b} = F$, leading to $\underline{c} = D$; and if $\underline{a} = F$, $\underline{b} = D$, $\neg \underline{b} = D$ and $\underline{c} = D$). Independent of our proof, (10) is an obvious result in PC. We shall use the symbol \underline{D} , as in r.h.s. of (10), to indicate that the l.h.s. of the equation leads to no definite consequence.

(c) A number (n) of implications in sequence

Consider

$$\underline{a}_1 \Rightarrow \underline{b}_1, \quad \underline{a}_2 \Rightarrow \underline{b}_2, \dots, \quad \underline{a}_n \Rightarrow \underline{b}_n \quad (11)$$

It is obvious that

$$\text{if } b_1 = a_2, \quad b_2 = a_3, \dots, \quad b_n = a_n \quad (12a)$$

$$\text{then } (11) \equiv (\underline{a}_1 \Rightarrow \underline{b}_n) \quad (12b)$$

On the other hand, between the $\frac{\theta}{\text{equations}}$ in (11),

$$\text{if } b_j = \neg a_{j+1}, \text{ for some } j = 1, 2, \dots, n-1 \quad (13a)$$

$$\text{then } (11) \equiv (\underline{a}_1 \rightarrow \underline{b}_n) = \underline{D} \quad (13b)$$

We shall not indicate the proof of this in any detail, but it readily follows from (10).

The same conditions (12a) are sufficient for the equations in (11) to be manipulated in the reverse sense. Thus, we obtain for the resultant

$$\begin{aligned} \neg b_n &\Rightarrow \neg a_n, \dots, \neg b_1 \Rightarrow \neg a_1 \\ &\equiv (\neg b_n \Rightarrow \neg a_1) \end{aligned} \quad (14)$$

It is to be noted that the r.h.s. of (14) is identical with (12b). Also, if (13a) holds, and

$$\neg b_{j+1} = a_j, \text{ for some } j = 1, 2, \dots, n-1 \quad (15a)$$

$$\text{then } \underbrace{(\neg b_n \Rightarrow \neg a_1)}_{\text{l.h.s of (14)}} \equiv (\neg b_n) \vee (\neg a_1) \quad (15b)$$

which, again, is identical with (13b).

Hence, for n implications in a sequence in PC, the net resultant is the same, irrespective of the direction in which the implications are applied — forward or reverse. This is not

the case in QPL, as we shall see later.

Thus, we have proved the condition for a pair of implications to lead to a valid resultant implication, and extended it for n implications. These results are obvious, and well understood, but the way in which they are presented here have an important application to the theory of our paper, applied to implications in statements of Type-1 in QPL.

3. Essentials of QPL needed for our derivations

(a) Sentences and states in EPL

In the four types of categorical statements A, E, I, O the standard QPL states \forall and \exists are used, with the implication being affirmative or negative. Thus, we may symbolically express the four forms of categorical statements by (16 a, b, c, d).

$$\underline{A} \equiv (\forall x) (\underline{a}(x) \Rightarrow \underline{b}(x)) , \quad \underline{E} \equiv (\forall x) (\underline{a}(x) \Rightarrow \neg \underline{b}(x)) \quad (16a, b)$$

$$\underline{I} \equiv (\exists x) (\underline{a}(x) \Rightarrow \underline{b}(x)) , \quad \underline{O} \equiv (\exists x) (\underline{a}(x) \Rightarrow \neg \underline{b}(x)) \quad (16c, d)$$

In these, the first two for A and E follow the standard practice

adopted traditionally for QPL. On the other hand, for I and O

we have made a small change, namely writing the PC statements

within the brackets that is quantified as $\underline{a}(x) \Rightarrow \underline{b}(x)$, instead

of $\underline{a}(x) \& \underline{b}(x)$ which is commonly used. The use of the logical

$\&$ is very significant in that the relation between \underline{a} and \underline{b}

under the existential quantifier \exists is a symmetric one. Thus,

"Some \underline{a} are \underline{b} " can also mean "Some \underline{b} are \underline{a} ". This is particularly

evident from the Venn diagram in Fig. 1(b) below. Therefore,

if we wish to emphasize the direction in which the implication

is applied, namely from \underline{a} to \underline{b} or from \underline{b} to \underline{a} we write the

same statement as

$$(\exists x) (\underline{a} \Rightarrow \underline{b}) \equiv (\exists x) (\underline{b} \Rightarrow \underline{a}) \quad (17)$$

In Eqn (17) we have omitted the variable x for \underline{a} and \underline{b} and

put it only for the quantifier $(\exists x)$. This practice will be

done throughout hereafter since all statements of Type-1 have the

same variable x for the terms inside the bracket that is quantified. The significant thing to be noticed about Eqn (17) is that we have in effect reversed the relation "Some \underline{a} are \underline{b} " into "Some \underline{b} are \underline{a} " and shown that the two are equivalent in the same sense as we have used for propositional calculus.

However, the reversal under the universal quantifier $(\forall x)$ is clearly different. As is very well understood, the PC sentence that is under the universal quantifier obeys all the rules of PC and therefore on reversing (16a), we will get the equation,

$$(\forall x) (\underline{a} \Rightarrow \underline{b}) \equiv (\forall x) (\neg \underline{b} \Rightarrow \neg \underline{a}) \quad (18)$$

We shall prove both (17) and (18) below when discussing the properties of the implication under quantification. It is clearly seen from (17) and (18) that the reversal of the PC sentence is different according as whether it is quantified by $(\forall x)$ or $(\exists x)$. As a consequence, in a sequence of two or more quantified

Implication, the reversed can produce a different result than the forward sequence. We will have many occasions to see this as we go along.

Before explaining the properties of the implication statements $(\forall x) (\underline{a} \Rightarrow \underline{b})$ and $(\exists x) (\underline{a} \Rightarrow \underline{b})$, we shall briefly mention the properties of the QPL statements in general. In attempting to obtain the Boolean algebraic representation of statements like those mentioned above, we found the need to have three basic states similar to the two basic states ($T = (1 \ 0)$, $F = (0 \ 1)$) for SNS logic. (See Ramachandran 1983 a, b).

Denoting a QPL vector by $(Y \ S \ E)$, the three basic states are

$$(1 \ 0 \ 0) = \text{"for all"}, \quad (0 \ 0 \ 1) = \text{"for none"}, \quad (0 \ 1 \ 0) = \text{"for some only and not for all or for none"} \quad (19a, b, c)$$

Putting the Boolean values 1 and 0 for each of $(Y \ S \ E)$, we obtain the total of $2 \times 2 \times 2 = 8$ different Boolean vectors that are possible in this algebra. On examination, the other five

states are found to have the logical equivalences

$$(1 \ 1 \ 0) = \text{"for all or some"} \equiv \text{"there exists"}, \quad (20a)$$

$$(0 \ 1 \ 1) = \text{"for none or some"} \equiv \text{"not for all"} \quad (20b)$$

$$(1 \ 0 \ 1) = \text{"for all or none"}, \quad (1 \ 1 \ 1) = \text{"indefinite"}, \\ (0 \ 0 \ 0) = \text{"impossible"} \quad (20c, d, e)$$

The general theory of the eight states in what may be called

extended predicate logic (EPL) is given in (Ramachandran 1983 a,b)

and applied to QPL sentences containing statements of the Type-2.

Here we will use only the two states $(\forall x)$ and $(\exists x)$ and will

obtain the other two states \emptyset (for none), and Λ (not for all)

by having negated statements under the quantifier. We shall not

also use the Boolean algebraic notation as ~~such~~ given by (19)

and (20) which was extensively employed for Type-2 statements.

Instead, we shall use the conventional logical formulae. But, as

we need the indefinite state (Δx) very often ^{and} since it occurs

frequently on combining two or more categorical statements, we shall

briefly discuss its properties here.

.23.

Just like the state $D = (1 \ 1)$ of SNS, which means

"either true or false", the state $\Delta = (1 \ 1 \ 1)$ of QPL

also has no information content and cannot be carried over in

the succeeding steps of ^a/_a connected series of statements. This

is best seen by writing it as,

$$(1 \ 1 \ 1) = (1 \ 1 \ 0) \text{ or } (0 \ 0 \ 1) = \begin{array}{l} \text{Either "there exists"} \\ \text{or "there does not exist"} \end{array} \quad (21)$$

We see that there is complete doubt about the quantifier, namely

"there exist"
whether it is /, or the negation of it, namely "there does not

exist". We can also ^{obtain} _{nnn} a (1 1 1) as the superposition of all

the three basic states in (19), but this interpretation is not

used in the present paper. As we shall illustrate by examples

later, the output of many pairs of statements of Type 1 containing

an implication leads to (Δx) when certain quantified inputs

are put in. Just as we ^{do with} _~ $\wedge \wedge \sim D$, we shall simply write the symbol

Δ at the conclusion to indicate that the conclusion ^{I does not have /} _X ^{P any} _L

informational value (instead of writing in full $(\Delta x)(\underline{a}(x) \Rightarrow \underline{b}(x))$).

(b) Outputs of single implication statements of Type-1 in QPL.

A summary of the eight independent input-output relations for implication of the form found in (16a) and (16c) are summarized in Table 2.

Table 2. Algorithms for obtaining the outputs of Type-1 statements of QPL

The proofs of these can be readily derived by drawing Venn diagrams of the corresponding sets which have the relevant relations between them. We shall only indicate the proof for Sl. Nos. 1 to 4 and 7, as they cover all essential ideas needed for the proof also for the remaining cases.

When the first one with Sl.No. 1 is translated into set theory, it says in effect — "In the universe of discourse if any member has the property \underline{a} , then it has the property \underline{b} ", and this is indicated in Fig.1(a). As will be seen from the figure, when all

Table 2. Algorithms for obtaining the outputs
of Type-1 statements of QPL

Sl. No.	Input	Statement	Output
		<u>Implies (\Rightarrow)</u>	
1	$(\forall x) (\underline{a})$	$(\forall x) (\underline{a} \Rightarrow \underline{b})$	$(\exists x) (\underline{b})$
2	$(\forall x) (\neg \underline{a})$	$(\forall x) (\underline{a} \Rightarrow \underline{b})$	D
3	$(\exists x) (\underline{a})$	$(\forall x) (\underline{a} \Rightarrow \underline{b})$	$(\exists x) (\underline{b})$
4	$(\exists x) (\neg \underline{a})$	$(\forall x) (\underline{a} \Rightarrow \underline{b})$	D
5	$(\forall x) (\underline{a})$	$(\exists x) (\underline{a} \Rightarrow \underline{b})$	$(\exists x) (\underline{b})$
6	$(\forall x) (\neg \underline{a})$	$(\exists x) (\underline{a} \Rightarrow \underline{b})$	D
7	$(\exists x) (\underline{a})$	$(\exists x) (\underline{a} \Rightarrow \underline{b})$	Δ
8	$(\exists x) (\neg \underline{a})$	$(\exists x) (\underline{a} \Rightarrow \underline{b})$	Δ and D
		<u>Reverse implies (\Leftarrow)</u>	
9	$(\forall x) (\underline{a})$	$(\forall x) (\underline{b} \Leftarrow \underline{a})$	$(\exists x) (\underline{b})$
10	$(\exists x) (\underline{a})$	$(\forall x) (\underline{b} \Leftarrow \underline{a})$	$(\exists x) (\underline{b})$
11	$(\forall x) (\underline{a})$	$(\exists x) (\underline{b} \Leftarrow \underline{a})$	$(\exists x) (\underline{b})$

Rules for reversal

$$(\forall x) (\underline{a} \Rightarrow \underline{b}) \equiv (\forall x) (\neg \underline{b} \Rightarrow \neg \underline{a}); (\exists x) (\underline{a} \Rightarrow \underline{b}) \equiv (\exists x) (\underline{b} \Rightarrow \underline{a})$$

$$(\forall x) (\underline{a} \Leftarrow \underline{b}) \equiv (\forall x) (\neg \underline{a} \Leftarrow \neg \underline{b}); (\exists x) (\underline{b} \Leftarrow \underline{a}) \equiv (\exists x) (\underline{a} \Leftarrow \underline{b})$$

the members of the set \underline{a} is exhausted, we cover only some (or all) of set \underline{b} , so that the output is only $(\exists x) (\underline{b})$. Obviously, $(\exists x)$ can also take care of the case when the set $\underline{a} = \text{set } \underline{b}$. The above property that $(\forall x) (\underline{a})$, when fed into $(\forall x) (\underline{a} \Rightarrow \underline{b})$, gives as output only $(\exists x) (\underline{b})$ (and not $(\forall x) (\underline{b})$) is a very important result and is vital for all the succeeding arguments.

As regards Sl. No. 2, we note from Fig 1(a) that points inside the square outside the shaded region, marked \underline{a} , can be either in the set \underline{b} or the set $\neg \underline{b}$. Therefore, $\neg \underline{a}$ can correspond to either \underline{b} or $\neg \underline{b}$, when $(\forall x) (\underline{a} \Rightarrow \underline{b})$, so that the output \underline{b} has the SNS state D, as indicated.

Considering Sl. No.3, if $(\exists x) (\underline{a})$ is the input, then we will have some part of the shaded area of set \underline{a} in Fig. 1(a) occupied. Clearly this corresponds to $(\exists x) (\underline{b})$ (as given in column 3 for this case), for the limiting case, when set $\underline{a} = \text{set } \underline{b}$

and all of set \underline{a} is included in the input, is also contained under the quantifier $(\exists x)$.

As regards Sl.No. 4, it can be verified from Fig. 1(a) that the input $(\exists x) (\neg \underline{a})$ can correspond to regions demarcated for set \underline{b} or set $\neg \underline{b}$, so that the output \underline{b} will be in the SNS state D, and the quantifier state $(\exists x)$. We denote this by the symbol D in the third column, indicating that \underline{b} cannot be carried over to further steps in any sequence of statements.

The outputs in Sl.Nos. 5 and 6 are immediately obvious from Fig. 2(a), following arguments like the above, and they are not discussed in detail.

The result, $(\Delta x) (\underline{b}(x))$ of Sl.No. 7 is understandable from Figs. 2(b) and 2(c). The sets \underline{a} and \underline{b} , with the same intersection $\underline{a} \& \underline{b}$ as in Fig. 2(a), are shown in Figs 2(b) and (c), with two different possibilities for the region of the diagram covered by

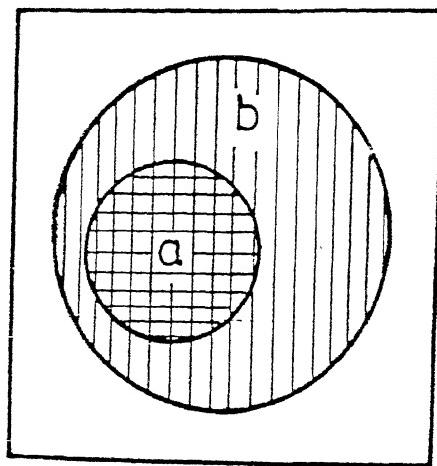
$(\exists x) (\underline{a})$. In Fig. 2(b), where the region $(\exists x) (\underline{a})$ is included in $\underline{a} \& \underline{b}$, the output is $(\exists x) (\underline{b})$; but if it is as in Fig. 2(c), then the region covered by the input $(\exists x) (\underline{a})$ is not included in $\underline{a} \& \underline{b}$, and the output is $\neg(\exists x) (\underline{b}) \equiv (\Phi' x) (\underline{b})$ (see Part II for the meaning of the symbol \equiv). Hence, for the full output \underline{b} , the quantifier is " $(\exists x)$ or (Φ') " = (1 1 1), as per Eqn. (21), which leads to the "indefinite" state (Δx) .

The result that the QPL output becomes $(\Delta x) (\underline{b})$ when both the implication and the input are in the $(\exists x)$ states, is very important for the discussions in the succeeding sections. The case of Sl.No. 8 is a combination of the conditions in Sl.Nos. 4 and 7, and hence we write the output as Δ and D , and it has no information value, for succeeding steps, if any, containing \underline{b} .

Fig.1 Venn diagrams for implications with $(\forall x)$, illustrating (a) $(\forall x) (\underline{a} \Rightarrow \underline{b})$, and (b) $(\forall x) (\neg \underline{b} \Rightarrow \neg \underline{a})$.

Fig.2 Venn diagrams for the implications $(\exists x) (\underline{a} \Rightarrow \underline{b}) \equiv (\exists x) (\underline{b} \Rightarrow \underline{a})$. (a) Existence of $\underline{a} \& \underline{b}$, (b) $(\exists x) (\underline{a}) \subset \underline{a} \& \underline{b}$, (c) $(\exists x) (\underline{a}) \not\subset \underline{a} \& \underline{b}$.

.29.

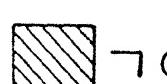
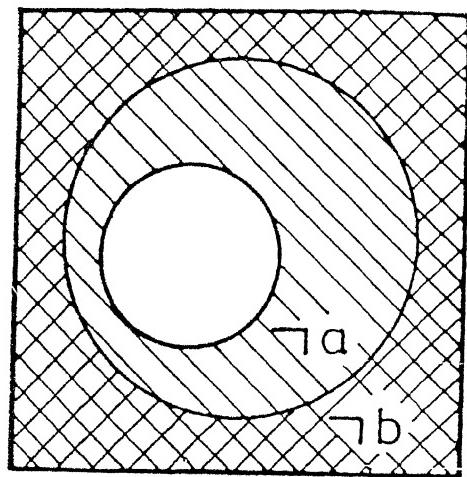


a



b

(a)



$\neg a$

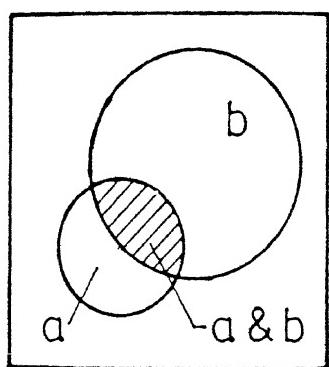


$\neg b$

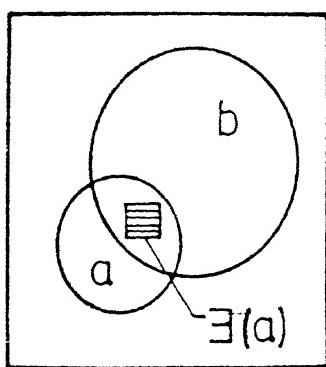
(b)

Fig.1

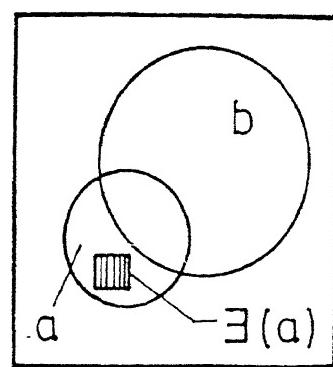
.29a.



(a)



(b)



(c)

Fig. 2

The last three lines in Table 2 corresponding to Sl.Nos.

9, 10 and 11 require no comment. They correspond to the reverse

operator of implication (\Leftarrow), and only the three cases where

a useful conclusion is obtained are listed, since all the rest

lead either to D or Δ or both. The implication is in the reverse

direction and it corresponds to the mathematical condition

"necessary", since the forward implies (\Rightarrow) corresponds to the

mathematical condition "sufficient".

(c) Reversal of quantified implications

Now, we shall examine the property of reversing an implication

from \underline{b} to \underline{a} in the case of Sl.No. 1. The Venn diagram in

Fig. 1(b) clearly gives the result $(\forall x) (\neg \underline{b} \Rightarrow \neg \underline{a})$, thus

confirming Eqn. (18) written earlier for the reversal of an

implication quantified by $(\forall x)$. This equivalence (18) is a well

established one in the literature of logic, and it is listed

at the bottom of Table 2.

In the same way, the equivalence (17) between $(\exists x) (\underline{a} \Rightarrow \underline{b})$ and $(\exists x) (\underline{b} \Rightarrow \underline{a})$ (also listed below Table 2) follows from Fig. 2(a), where both of them are consequences of the fact that $(\exists x) (\underline{a} \& \underline{b})$ is true. This equivalence for $(\exists x)$ is not commonly used and is not found in the literature, but it is extremely important for deriving the validity of categorical syllogisms (see Section 4). It will be noticed that the reversed SNS statement inside the bracket quantified by $(\exists x)$ is different from that quantified by $(\forall x)$. Because of this, when a pair of QPL statements, one with $(\forall x)$ and the other with $(\exists x)$, is reversed, only one or the other leads to a non-D result. We shall discuss this further in subsection 3(d) and section 4.

(d) Conclusions from two categorical statements

In this subsection, we shall combine two quantified statements $(\mathcal{Q}_1 x) (\underline{a} \Rightarrow \underline{b})$, $(\mathcal{Q}_2 x) (\underline{b} \Rightarrow \underline{c})$ and work out the resultant, for $\mathcal{Q}_1, \mathcal{Q}_2 = \forall \text{ or } \exists$ and $\underline{b}' = \underline{b} \text{ or } \neg \underline{b}$. The results are

summarized in Table 3. The first entry appears to be a simple

Table 3. Sequence of two implications in succession connecting the terms \underline{a} and \underline{c} via \underline{b}

extension of the PC equation $(\underline{a} \Rightarrow \underline{b}), (\underline{b} \Rightarrow \underline{c}) = (\underline{a} \Rightarrow \underline{c})$

to the universal quantifier state $(\forall x)$. However, this is not such a trivial result, as its proof will show. Thus,

$$\begin{aligned} \text{For } (\forall x) (\underline{a} \Rightarrow \underline{b}), (\forall x) (a_T) \mapsto (\exists x) (b_T) \\ \text{and } (\forall x) (a_F) \mapsto (\exists x) (b_D) \end{aligned} \quad (22a)$$

$$\begin{aligned} \text{For } (\forall x) (\underline{b} \Rightarrow \underline{c}), (\exists x) (b_T) \mapsto (\exists x) (c_T) \\ \text{and } (\exists x) (b_D) \mapsto (\exists x) (c_D) \end{aligned} \quad (22b)$$

Combining the two, we get, for the relation between \underline{a} and \underline{c} , the correspondences

$$(\forall x) (a_T) \mapsto (\exists x) (c_T), (\forall x) (a_F) \mapsto (\exists x) (c_D) \quad (23a)$$

which is identical in nature with the r.h.s of (22a) of $(\forall x) (\underline{a} \Rightarrow \underline{b})$. Hence the correspondences in (23a) yield $(\forall x) (\underline{a} \Rightarrow \underline{c})$, and we can write the analog of (8a) as below.

$$(\forall x) (\underline{a} \Rightarrow \underline{b}), (\forall x) (\underline{b} \Rightarrow \underline{c}) = (\forall x) (\underline{a} \Rightarrow \underline{c}) \quad (23b)$$

Table 3. Sequence of two implications in succession
connecting the terms \underline{a} and \underline{c} via \underline{b} .

Sl. No.	First statement	Second Statement	Conclusion
1	$(\forall x) (\underline{a} \Rightarrow \underline{b})$	$(\forall x) (\underline{b} \Rightarrow \underline{c})$	$(\forall x) (\underline{a} \Rightarrow \underline{c})$
2	$(\forall x) (\underline{a} \Rightarrow \neg \underline{b})$	$(\forall x) (\underline{b} \Rightarrow \underline{c})$	\mathcal{D}
3	$(\exists x) (\underline{a} \Rightarrow \underline{b})$	$(\forall x) (\underline{b} \Rightarrow \underline{c})$	$(\exists x) (\underline{a} \Rightarrow \underline{c})$
4	$(\exists x) (\underline{a} \Rightarrow \neg \underline{b})$	$(\forall x) (\underline{b} \Rightarrow \underline{c})$	\mathcal{D}
5	$(\forall x) (\underline{a} \Rightarrow \underline{b})$	$(\exists x) (\underline{b} \Rightarrow \underline{c})$	Δ
6	$(\forall x) (\underline{a} \Rightarrow \neg \underline{b})$	$(\exists x) (\underline{b} \Rightarrow \underline{c})$	\mathcal{D} and Δ
7	$(\exists x) (\underline{a} \Rightarrow \underline{b})$	$(\exists x) (\underline{b} \Rightarrow \underline{c})$	Δ
8	$(\exists x) (\underline{a} \Rightarrow \neg \underline{b})$	$(\exists x) (\underline{b} \Rightarrow \underline{c})$	\mathcal{D} and Δ

This corresponds to the first mood in the first figure

(named Barbara) of the syllogisms in traditional logic (see next section 4).

Considering the entry for Sl.No. 2, the first implication gives the output $(\exists x) (\neg \underline{b})$ and this, going into the second implication, gives $(\exists x) (c_D)$ as the final output, thus leading to the conclusion $\sim D$.

Sl.No. 3 of Table 3 is very interesting. $(\exists x) (\underline{a})$ as input to the first statement gives $(\exists x) (\underline{b})$ as output, and this, going as input into $(\forall x) (\underline{b} \Rightarrow \underline{c})$ gives the output $(\exists x) (\underline{c})$, according to Sl.No. 3 of Table 2. Hence, the two statements together yield the conclusion $(\exists x) (a \Rightarrow c)$, and we obtain Eqn.(24).

$$(\exists x) (\underline{a} \Rightarrow \underline{b}), (\forall x) (\underline{b} \Rightarrow \underline{c}) = (\exists x) (\underline{a} \Rightarrow \underline{c}) \quad (24)$$

This combination of statements is also in the first figure of the syllogisms in traditional logic, and bears the name Darii (see Section 4).

All the rest of the combinations of two implications in Table 3 lead to either \underline{D} or Δ . (we use the heavy letter symbol \underline{D} (and not D), for it refers to the relation $\underline{a} \underline{D} = \underline{c}$ which yields c_D for both inputs a_T and a_F (see Ramachandran, 1981).

Of these, Sl.No. 5 is particularly interesting, for it has the quantifier sequence (\forall, \exists) and yet leads to the QPL state Δ . This is because $(\forall x) (\underline{a} \Rightarrow \underline{b})$ has the output $(\exists x) (\underline{b})$, and this, going into $(\exists x) (\underline{b} \Rightarrow \underline{c})$ will lead only to a Δ -state output as we have particularly pointed out in the last subsection (see Sl.No. 7 of Table 2). The same reason makes Sl.No. 7 of Table 3 also have a Δ -state conclusion. The occurrence of the quantifier state Δ for the resultant of two QPL statements under suitable conditions also plays an important part in the general theory of categorical syllogisms, considered in the next section.

Thus, we have obtained the remarkable conclusion that there is, in essence, only two forms of a pair of categorical statements, in the nature of A, E, I or O, leading to valid conclusions, which are independent, namely Eqns (23b) and (24), listed as Sl.Nos. 1 and 3 in Table 3. We shall examine in the next section the 64 different ways (or moods) in which these statements can combine, as discussed in traditional logic, and indicate their validity, by bringing them to a form isomorphous with one of these two, which also will enable us to obtain the nature of the conclusion. Since the reverse of such an ordered pair of statements is, in general, not equivalent to the forward one, the test must be made with both the forward and reverse forms, and if either gives a valid conclusion, that should be taken as the conclusion of the pair. We shall prove in the next section that no ambiguity arises in this procedure. But, it can happen that both forward and reverse senses may lead

to Δ or Δ' as conclusions, in which case the pair is invalid (see Section 4 (C)).

One more general result for QPL statements of Type-1 remains to be stated, since it will be applied in the treatment of Section 4. This is that

$$(\forall x) (\underline{a} \Rightarrow \underline{b}) \supset (\exists x) (\underline{a} \Rightarrow \underline{b}) \quad (25)$$

Thus, it can happen, as with $(\forall x) (\underline{s} \Rightarrow \neg \underline{m})$, $(\forall x) (\underline{m} \Rightarrow \underline{p})$, that the pair leads only to the conclusion Δ both in the forward and in the reverse directions. In such a case, we can take the reverse of the first statement to obtain $(\forall x) (\underline{m} \Rightarrow \neg \underline{s})$, and then apply (25) to this to obtain $(\exists x) (\underline{m} \Rightarrow \neg \underline{s})$. If we reverse it again to get $(\exists x) (\neg \underline{s} \Rightarrow \underline{m})$, we are led to the pair of statements in the l.h.s of (26), which have a conclusion, by isomorphism with Eqn (24). We put $\neg \underline{s} = \underline{a}$, $\underline{m} = \underline{b}$ and $\underline{p} = \underline{c}$ in l.h.s. of (24) and obtain Eqn. (26)

$$(\exists x) (\neg \underline{s} \Rightarrow \underline{m}), (\forall x) (\underline{m} \Rightarrow \underline{p}) = (\exists x) (\neg \underline{s} \Rightarrow \underline{m}) \quad (26)$$

The resultant syllogism, namely $(\forall x) (\underline{m} \Rightarrow \underline{p})$,

$(\forall x) (\underline{s} \Rightarrow \neg \underline{m}) = (\exists x) (\neg \underline{s} \Rightarrow \underline{m})$ is one that is not found in the list of eighteen in traditional logic and the pair in the l.h.s. is usually listed as invalid, for instance, in Delong (1971). Our technique has revealed a number of such new valid syllogisms, and the next section gives the details of the applications of the results presented in this subsection to the 64 possible moods.

4. Theory of categorical syllogisms and the newly added examples.

(a) General principles

We shall not give an introduction to categorical syllogisms, as they are well-understood in traditional logic (see e.g. Church, 1966; Delong, 1971). It connects a minor term \underline{s} to a major term \underline{p} via the middle term \underline{m} , using two categorical statements — namely the minor premise connecting \underline{p} and \underline{m} and the major premise connecting \underline{s} and \underline{m} . The two categorical

statements can be connected together in four different ways,

and the name "figure" is used for these, following Aristotle.

We have given the direction of the implication in each of the

four figures that are possible ~~at the bottom~~ⁱⁿ of Table 4. Following

tradition, we use only the four forms A, E, I, O for each of

the major and minor premises, so that there are 16 possible

"moods" in each figure, making a total of 64 in all the four

figures.

All the moods can be examined for validity, using the following criteria:

(i) If both the major and the minor premises in the syllogism are I or O, then the two statements have the pattern \exists, \exists of Sl.Nos. 7 or 8 of Table 3 and the resultant is D or A, leading to an "invalid" conclusion.

(ii) If one is A or E, and the other I or O, so that we have the sequence \forall, \exists or \exists, \forall , ^{and} if it is not in isomorphism

with $(\exists x) (\underline{a} \Rightarrow \underline{b})$, $(\forall x) (\underline{b} \Rightarrow \underline{c})$ we attempt to obtain it

by reversing one, or both, the statements. We shall show below

that at most one of these possibilities leads to a valid conclusion,

and that conclusion is then assigned to the syllogism. If none

are valid, the syllogism is taken to have no valid conclusion,

and to lead always to indefiniteness.

σ^r

(iii) If both are $A \not\propto E$, and we have the pattern \forall_1, \forall_2 , we test if it is in isomorphism with $(\forall x) (\underline{a} \Rightarrow \underline{b})$, $(\forall x) (\underline{b} \Rightarrow \underline{c})$.

If not, ^{W^v} try reversing one, or both, the statements, so as to

obtain four possibilities in all. We shall show that either

all four are invalid, or two are valid and two invalid. If two

are valid, then the two valid conclusions will be equivalent

under reversal, so that there is no ambiguity in the conclusion.

(iv) If, in (iii), all four are invalid, we obtain, using the analog of Eqn. (25), one of the two possibilities — \exists_1, \forall_2 or \forall_1, \exists_2 . We then follow the procedure in (ii) for the four

rearrangements of each. Then, it will be found that one, and only one, possibility (of the eight) is valid and all the rest are invalid. It will not happen that all eight are invalid in this case.

We shall first discuss the results obtained by using (i) to. (iv) in subsections (b) and (c) and prove these in subsection (d).

(b) Table of categorical syllogisms

If the procedures (i), (ii), (iii), (iv) of the previous subsection are applied, as appropriate, to the 64 possible pairs of statements of the forms A, E, I, O, 18 of the 19 traditional moods (as listed e.g. by Church (1966) or Delong (1971)) are obtained, provided we use, for the SNS statement f the conclusion, only either $\underline{s} \Rightarrow p$ or $\underline{s} \Rightarrow \neg \underline{p}$, i.e. the conclusion is also one of the four forms A, E, I, O. These are given in Table 4 on the left hand side. Following standard practice, the

symbols A, E, I or O of the major premise, minor premise, and conclusion are given in serial order, followed by the number (1, 2, 3 or 4) of the figure, for each valid mood. The mnemonic latin words for each mood is also given, which has its three vowels in the same order as the symbol. We shall give, in subsection 4(c) the demonstration of typical examples.

Table 4. Symbols and Names of Valid Moods of the Old and New Types

When this was done for all the 64 combinations, quite a number were found to have conclusions with implication $\neg s \Rightarrow p$ and $\neg s \Rightarrow \neg p$, with, of course, either ($\forall x$) or ($\exists x$) as quantifier. Thus, they have one of the forms A', E', I', O' (shown in (2a-d)) as conclusion. Therefore, these were systematically searched for, and then 18 more valid syllogisms were discovered in the moods with symbols as listed in the right hand side of Table 4. The mnemonic words for these were coined and they are also listed in Tab.e 4. They all have four vowels, the first three corresponding to the forms of major premise, minor premise and conclusion, while the occurrence of the fourth vowel indicates that the syllogism is of the new type and has A', E', I' or O' as conclusion.

Table 4: Symbols and Names of Valid Moods of the
Old and New Types

Old Type	New Type
<u>Figure 1 : m → p, s → m, s → p</u>	
<u>AAA-1</u> , Barbara	<u>AEI'-1</u> , Kaveriar
<u>AII-1</u> , Darii	<u>EEO'-1</u> , Electronic
<u>EAE-1</u> , Celarent	<u>IEI'-1</u> , Dimension
<u>EIO-1</u> , Ferio	<u>OEO'-1</u> , Ponderous
<u>Figure 2 : p → m, s → m, s → p</u>	
<u>AEE-2</u> , Camestres	<u>AAO'-2</u> , Bangalore
<u>AOO-2</u> , Baroco	<u>EEO'-2</u> , Neelotpal
<u>EAE-2</u> , Cesare	<u>IEI'-2</u> , Nivedita
<u>EIO-2</u> , Festino	<u>OAI'-2</u> , Moravia
<u>Figure 3 : m → p, m → s, s → p</u>	
<u>AAI-3</u> , Darapti	<u>AEI'-3</u> , Magnesia
<u>AII-3</u> , Datisti	<u>AOI'-3</u> , Sarojini
<u>EAO-3</u> , Felapton	<u>EEO'-3</u> , Development
<u>EIO-3</u> , Feriso	<u>EOO'-3</u> , Mesozoic
<u>IAI-3</u> , Disamis	<u>IEI'-3</u> , Indefinite
<u>OAO-3</u> , Bocardo	<u>OEO'-3</u> , Lodestone
<u>Figure 4: p → m, m → s, s → p</u>	
<u>AEE-4</u> , Calemes	<u>AAE'-4</u> , Management
<u>EAO-4</u> , Fesapo	<u>EEO'-4</u> , Generous
<u>EIO-4</u> , Fresison	<u>EOO'-4</u> , Leonora
<u>IAI-4</u> , Dimatis	<u>IEI'-4</u> , Silesia

(c) Illustrations of the procedures (i) to (iv) of Section 4(a)

We shall first give a few examples for the standard syllogisms in the four figures corresponding to the left half of Table 4, and then deal with one or two in the new syllogisms in the right half.

Considering the first figure, Barbara and Darii are the standard ones worked out in Section 3, with one of which all others are sought to be brought into isomorphism. We shall work out Celarent (EAE-1) which is straightforward. In this, EA corresponds to the sequence of statements on the l.h.s. of (27a), which gives again an E-statement from \underline{s} to $\neg \underline{p}$, leading to the listing EAE-1.

$$(\forall x) (\underline{s} \Rightarrow \underline{m}), (\forall x) (\underline{m} \Rightarrow \neg \underline{p}) = (\forall x) (\underline{s} \Rightarrow \neg \underline{p}) \quad (27a)$$

On reversing both statements in (27a), we obtain the l.h.s. of (27b), which gives $(\forall x) (\underline{p} \Rightarrow \neg \underline{s})$ as conclusion. This is

equivalent to the r.h.s. of (27a).

$$\begin{aligned} (\forall x) (\underline{p} \Rightarrow \neg \underline{m}), (\forall x) (\neg \underline{m} \Rightarrow \neg \underline{s}) \\ = (\forall x) (\underline{p} \Rightarrow \neg \underline{s}) \end{aligned} \quad (27b)$$

On the other hand, reversing only one of the statements on the l.h.s of (27a) leads to $(\forall x) (\underline{s} \Rightarrow \underline{m})$, $(\forall x) (\underline{p} \Rightarrow \neg \underline{m})$ and $(\forall x) (\neg \underline{m} \Rightarrow \neg \underline{s})$, $(\forall x) (\underline{m} \Rightarrow \neg \underline{p})$ which are disconnected, and cannot lead to a valid conclusion. This illustrates (iii), with the sequence \forall_1, \forall_2 which has a valid conclusion.

From the examples in the second figure, we may take Baroco (A00-2). Its two statements are $(\forall x) (\underline{p} \Rightarrow \underline{m})$, $(\exists x) (\underline{s} \Rightarrow \neg \underline{m})$ which are disconnected. However, it can be made to be ^a/connected pair by reversing the first statement alone, when we obtain (28a) with a valid resultant.

$$(\exists x) (\underline{s} \Rightarrow \neg \underline{m}), (\forall x) (\neg \underline{m} \Rightarrow \neg \underline{p}) = (\exists x) (\underline{s} \Rightarrow \neg \underline{p}) \quad (28a)$$

The other two possibilities obtained by reversal of one or the

.46.

other of the A,0 sequence yield only the disconnected pair

$(\forall x) (\underline{p} \Rightarrow \underline{m})$, $(\exists x) (\underline{s} \Rightarrow \neg \underline{m})$, *or one with* *the resultant D and A,*
as in (28b).

$$(\forall x) (\underline{p} \Rightarrow \underline{m}), (\exists x) (\neg \underline{m} \Rightarrow \underline{s}) = \underline{D} \text{ and } \Delta \quad (28b)$$

This illustrates (ii) with the sequence A, E, which has a valid conclusion.

As an illustration of an invalid pair of statements (but which is not E, E as in (i)), we shall take the example A I in the fourth figure, which is

$$(\forall x) (\underline{p} \Rightarrow \underline{m}), (\exists x) (\underline{m} \Rightarrow \underline{s}) = \Delta \quad (29a)$$

Although the SNS statements are connected, the quantifier sequence is an A followed by a E, which leads to the indefinite state D. We give the three other combinations obtained by reversing one, or the other, or both the statements, below in (29 b,c,d), all of which also lead to no valid conclusion.

$$(\forall x) (\neg \underline{m} \Rightarrow \neg \underline{p}), (\exists x) (\underline{m} \Rightarrow \underline{s}) \quad (\text{Disconnected}) \quad (29b)$$

$$(\forall x) (\underline{p} \Rightarrow \underline{m}), (\exists x) (\underline{s} \Rightarrow \underline{m}) \quad (\text{Disconnected}) \quad (29c)$$

$$(\exists x) (\underline{s} \Rightarrow \underline{m}), (\forall x) (\neg \underline{m} \Rightarrow \neg \underline{p}) = \text{D} \quad (29d)$$

For an illustration of (iv), we may point out the example

of $\underline{\underline{A}} \underline{\underline{E}}$ in the first figure, which was discussed at the end of Section 3(c). It is invalid as is seen from (30a).

$$(\forall x) (\underline{s} \Rightarrow \neg \underline{m}), (\forall x) (\underline{m} \Rightarrow \underline{p}) = \text{D} \quad (30a)$$

The reversal of its terms also gives only invalid conclusions,

as is readily verified. But, on applying Eqn (25) and obtaining

an $(\exists x)$ for the first statement, we deduce the valid conclusion

$(\exists x) (\neg \underline{s} \Rightarrow \underline{p}) (\underline{I}'')$ as shown earlier in (26). It can be

readily verified that the other connected combination that is

possible, leads only to D or Δ , as in (30b) below:

$$(\forall x) (\neg \underline{p} \Rightarrow \neg \underline{m}), (\exists x) (\underline{m} \Rightarrow \neg \underline{s}) = \text{D} \text{ and } \Delta \quad (30b)$$

On obtaining $(\exists x) (\underline{m} \Rightarrow \underline{p})$ from the second statement in (30a), we get the connected pair in the l.h.s. of (30c), which again is \underline{D} or Δ . On reversing both the statements on the l.h.s. of (30c), we obtain (30d), which has, however, the conclusion Δ , since the sequence of quantifiers in \exists, \forall . The remaining four possibilities are disconnected, and therefore do not lead to a valid conclusion.

$$(\forall x) (\underline{s} \Rightarrow \neg \underline{m}), (\exists x) (\underline{m} \Rightarrow \underline{p}) = \underline{D} \text{ and } \Delta \quad (30c)$$

$$(\exists x) (\underline{p} \Rightarrow \underline{m}), (\forall x) (\underline{m} \Rightarrow \neg \underline{s}) = \Delta \quad (30d)$$

Thus, we find that $\overset{\wedge}{\underline{A} \underline{E}}$ in the first figure has only the conclusion I'. In fact, this syllogism (AEI'-1) was one of the first examples of the new type of syllogism that was discovered during our studies, and it has been given the name Kaveriar ('Kaveri' being an important river close to Bangalore and 'ar' being the termination for 'river' in Tamil).

It may be mentioned that, among the four arrangements obtained by reversals of two categorical statements ($\forall_1 x$) ($\underline{a} \Rightarrow \underline{b}'$) and ($\forall_2 x$) ($\underline{b} \Rightarrow \underline{c}$), two will not form a connected chain from \underline{a} to \underline{c} via the middle term \underline{b} , so that only two need be examined. This is seen, for instance, in (27a, b) and (28a, b). Also, in (30a, b) and (30c, d), this feature has been taken advantage of, and only two possible arrangements, which are reverses of each other, are examined. Although the manipulations in (27) to (30) appear to be complicated, the proof of the uniqueness for the conclusion given by a pair of statements connecting \underline{s} to \underline{p} , via the middle term \underline{m} , is quite simple. We give the proof briefly in the next subsection.

(d) Proof of uniqueness of the procedures (i)^{b3}-(iv) for obtaining the resultant.

The proof is not demonstrative, but existential. We assume, in each case, that one type of rearrangement (after reversal)

leads to a valid conclusion. We then show that no other

non-equivalent conclusion follows by any other possible rearrangement

In doing this, we consider only the other connected rearrangements

i.e which have either $a \rightarrow m, m \rightarrow b$ or $b \rightarrow m, m \rightarrow a$.

(i) \exists, \exists : In this case, no valid conclusion is possible, for the resultant quantifier is always Δ . Hence, there can be no duplication.

(ii) \forall, \exists : Here again, if there is no valid resultant, there is no question of duplication. Therefore, we assume that there is one valid arrangement, in which \exists must precede \forall , as in

$$(\exists x) (\underline{\underline{a}} \Rightarrow \underline{\underline{k}}), (\forall x) (\underline{\underline{l}} \Rightarrow \underline{\underline{b}}) \quad (31a)$$

Note that, for validity, $\underline{\underline{k}}$ must be equal to $\underline{\underline{l}}$, i.e.

$$\underline{\underline{k}} = \underline{\underline{l}} = \text{either } \underline{\underline{m}} \text{ for both, or } \neg \underline{\underline{m}} \text{ for both} \quad (31b)$$

Now, on reversing both statements (which is a necessary condition

for connectedness), we obtain,

$$(\forall x) (\neg \underline{b} \Rightarrow \neg \underline{l}), (\exists x) (\underline{k} \Rightarrow \underline{a}) = \underline{D} \text{ and } \Delta \quad (31c)$$

The Δ arises from the sequence \forall, \exists (see Table 3, Sl.Nos.5, 6)

and \underline{D} arises because $\underline{k} = \neg \underline{l}$ if (31b) is satisfied.

The other two rearrangements will be disconnected, and hence the valid conclusion is unique.

(iii, \forall, \forall and conclusion exists: In this case, assuming connectedness to exist, there is only one other rearrangement possible for reversal. Hence, if

$$(\forall x) (\underline{a} \Rightarrow \underline{m}), (\forall x) (\underline{m} \Rightarrow \underline{b}) = (\forall x) (\underline{a} \Rightarrow \underline{b}) \quad (32a)$$

then

$$\begin{aligned} (\forall x) (\neg \underline{b} \Rightarrow \neg \underline{m}), (\forall x) (\neg \underline{m} \Rightarrow \neg \underline{a}) &= \\ &= (\forall x) (\neg \underline{b} \Rightarrow \neg \underline{a}) \end{aligned} \quad (32b)$$

and the two conclusions in (32a) and (32b) are equivalent. The other two rearrangements are disconnected and hence invalid.

Therefore, in this case, although there is duplication, there is

no ambiguity.

(iv) \forall, \forall and no valid conclusion : Since there is no conclusion, the two statements have the pattern

$$(\forall_1 x) (\underline{a} \Rightarrow \underline{k}), (\forall_2 x) (\underline{\ell} \Rightarrow b) \text{ with } \underline{k} = \neg \underline{\ell} \quad (33)$$

Straightaway, obtaining an $(\exists x)$ statement from either \forall_1 or \forall_2 ,

and rearranging, after reversal, for connectedness, we obtain the four possibilities — the other four are disconnected:

$$(\exists_1 x) (\underline{a} \Rightarrow \underline{k}), (\forall_2 x) (\underline{\ell} \Rightarrow \underline{b}) = \textcircled{D} \quad (34a)$$

$$(\forall_2 x) (\neg \underline{b} \Rightarrow \neg \underline{\ell}), (\exists_1 x) (\underline{k} \Rightarrow \underline{a}) = \Delta \quad (34b)$$

$$(\forall_1 x) (\underline{a} \Rightarrow \underline{k}), (\exists_2 x) (\underline{\ell} \Rightarrow \underline{b}) = \textcircled{D} \text{ and } \Delta \quad (34c)$$

$$\begin{aligned} (\exists_2 x) (\underline{b} \Rightarrow \underline{\ell}), (\forall_1 x) (\neg \underline{k} \Rightarrow \underline{a}) &= (\exists x) (\underline{b} \Rightarrow \underline{a}) \quad (34d) \\ &= (\exists x) (\underline{a} \Rightarrow \underline{b}) \end{aligned}$$

It is seen that, with the condition $\underline{k} = \neg \underline{\ell}$ in (33), only one of the four, viz (34d), is valid, and the others are invalid.

By putting (iii) and (iv) together, we can say that the

sequence \forall, \forall for a pair of categorical statements always has a conclusion, either with the universal quantifier \forall , or (if this is not possible) with the existential quantifier \exists .

Because of the proof of uniqueness of the valid conclusion of a pair of categorical statements, in searching for the conclusion of a given pair, if a valid conclusion is arrived at, we need not check any further rearrangements.

(e) Summary of Section 4

It will be noticed that although the categorical syllogisms covered by Table 4 in section 4(b) have only statements of the form A, E, I, O for the two premises, the theory in section 4(a) and its proof in section 4(d) make no such restrictions, and the two statements considered in these can have all the eight forms A,E, I,O, A', E', I', O' for them. Thus, quite generally, there are $8 \times 8 = 64$ moods in each figure, leading in all to 256

possibilities. However, we are not considering all these pairs of statements, since the procedures (i) to (iv) can always be used to check for their validity, and to find the resultant when any pair is valid. In effect, the procedures (i) to (iv) have the following consequences:

(α) If the pair has only \underline{A} , \underline{E} , \underline{A}' , \underline{E}' (i.e. of the type \forall, \forall), then it always has a conclusion, either of the type \forall , or of the type \exists .

(β) If it has only \underline{I} , \underline{O} , \underline{I}' , \underline{O}' (i.e. of the type \exists, \exists), then it is invalid and has no definite conclusion.

(γ) For the rest (of the type \exists, \forall or \forall, \exists), some are valid, and some invalid. When valid, the conclusion is always of the type \exists , i.e. one of \underline{I} , \underline{O} , \underline{I}' , \underline{O}' .

The above theory can be extended to a connected sequence of n categorical statements, the QPL analog of the SNS sequence considered in Eqn (11) of Section 2(e). As the study of these has not yet been finalized, this is only presented in outline in the next section.

5. Resultant of n connected categorical statements

(a) Three possible patterns leading to conclusions

Suppose we have a sequence of n connected statements

$$(\forall x) (\underline{a} \Rightarrow \underline{b}_1), (\forall x) (\underline{a}_2 \Rightarrow \underline{b}_2), \dots (\forall x) (\underline{a}_j \Rightarrow \underline{b}_j) \dots (\forall x) (\underline{a}_n \Rightarrow \underline{p}) \quad (35a)$$

the condition for connectedness being that

$$\underline{b}_j = \underline{a}_{j+1}, \text{ or } \neg \underline{b}_j = \underline{a}_{j+1}, j = 1 \text{ to } (n-1) \quad (35b)$$

and

$$\forall_j = \forall \text{ or } \exists, j = 1 \text{ to } n \quad (35c)$$

since we deal only with categorical statements of the forms

A, E, I, O, A', E', I', O'. We divide our discussion into two

categories — (i) All quantifiers are $(\forall x)$ and (ii) Some are $(\exists x)$ and the remaining $(\forall x)$. The third case when all are $(\exists x)$

need not be considered, since even two implications in succession, both $(\exists x)$, will lead to (Δx) as consequence, so that there is no valid conclusion when all are $(\exists x)$.

(b) Analogue of Barbara

(i) All \forall_j are \forall : In this case, by repeated application of

Barbara (Sl. No. 1 of Table 3) we obtain, using (35b),

$$\begin{aligned} (\forall x) (\underline{s} \Rightarrow \underline{b}_1), (\forall x) (\underline{b}_1 \Rightarrow \underline{b}_2), \dots, (\forall x) (\underline{b}_j \Rightarrow \underline{b}_{j+1}) \quad (36a) \\ \dots, (\forall x) (\underline{b}_{n-1} \Rightarrow \underline{p}) = (\forall x) (\underline{s} \Rightarrow \underline{p}) \end{aligned}$$

and the necessary conditions in (35a) are that $\underline{b}_j = \underline{a}_{j+1}$ for

all j . It is readily verified that, on reversing the whole of

the l.h.s. of (36), we obtain again a connected sequence

$$\begin{aligned} (\forall x) (\neg \underline{p} \Rightarrow \neg \underline{b}_{n-1}), (\forall x) (\neg \underline{b}_{n-1} \Rightarrow \neg \underline{b}_{n-2}), \dots, \\ (\forall x) (\neg \underline{b}_{j+1} \Rightarrow \neg \underline{b}_j), \dots, (\forall x) (\neg \underline{b}_1 \Rightarrow \neg \underline{s}) \quad (36b) \\ (\forall x) (\neg \underline{p} \Rightarrow \neg \underline{s}) \end{aligned}$$

and the conclusion in (36b) is the equivalent, under reversal,

of that in (36a). Thus, the resultant in r.h.s. of (36a) is unique.

(c) Two analogues of Darii in the forward and reverse directions.

(ii) Some \forall_j are \exists , others \forall : In this case, there cannot be two \exists 's in succession, since it leads to a (Δ) and the

resultant conclusion of all the statements also will be in the Δ -state and hence invalid. Also, a sequence \forall, \exists cannot occur anywhere, since this also leads to a Δ -state (as per Sl.Nos 5 and 6 of Table 3). Hence, only \forall_1 or \forall_n can be \exists .

(d) In the former case, the forward sequence (35a) is implementable, with the additional condition $b_j = a_{j+1}$ for all $j = 1$ to $(n-1)$, by repeated application of Darii (Sl.No.3 of Table 3), and the conclusion is $(\exists x) (\underline{s} \Rightarrow \underline{p})$. For ready reference, we give this, in (37a):

$$(\exists x) (\underline{s} \Rightarrow \underline{b}_1), (\forall x) (\underline{b}_1 \Rightarrow \underline{b}_2), \dots, (\forall x) (\underline{b}_{n-1} \Rightarrow \underline{p}) \equiv (\exists x) (\underline{s} \Rightarrow \underline{p}) \quad (37a)$$

In this case, reversing the l.h.s. of (37a) leads to Δ -state for the conclusion because of the sequence \forall, \exists at the end; but this does not matter because by our theory, only one of the forward and the reversed sequences need have a definite conclusion

for the set of categorical statements to have a valid resultant.

(β) However, in the reverse direction, \exists can be the quantifier of the first statement. So, we can look for a resultant for a sequence of n statements, as in (36a) and (37a), but with $(\exists x)$ quantifying the last one, under suitable conditions connecting \underline{b}_j and \underline{a}_{j+1} . Let us write it in the forward direction as follows:

$$(\forall x) (\underline{s} \Rightarrow \underline{b}_1), (\forall x) (\underline{b}_1 \Rightarrow \underline{b}_2), \dots, (\forall x) (\underline{b}_{n-2} \Rightarrow \underline{b}_{n-1}) \\ (\exists x) (\underline{a}_{n-1} \Rightarrow \underline{p}) \quad (38a)$$

The conclusion of this is a Δ -state; but when written in the reverse direction, it takes the form

$$(\exists x) (\underline{p} \Rightarrow \underline{a}_{n-1}), (\forall x) (\neg \underline{b}_{n-1} \Rightarrow \neg \underline{b}_{n-2}), \dots, \\ (\forall x) (\neg \underline{b}_2 \Rightarrow \neg \underline{b}_1), (\forall x) (\neg \underline{b}_1 \Rightarrow \neg \underline{s}) \quad (38b)$$

This is seen to have a form isomorphous with (37a) if

by homology,

$\underline{a}_{n-1} = \neg \underline{b}_{n-1}$, and the resultant is $(\exists x) (\underline{p} \Rightarrow \neg \underline{s})$ / which is equivalent to $(\exists x) (\neg \underline{s} \Rightarrow \underline{p})$. Hence, we obtain the

third category of a sequence of n statements with a conclusion —

namely

$$(\forall x) (\underline{s} \Rightarrow \underline{b}_1), (\forall x) (\underline{b}_1 \Rightarrow \underline{b}_2), \dots, (\forall x) (\underline{b}_{n-2} \Rightarrow \underline{b}_{n-1}), \\ (\exists x) (\neg \underline{b}_{n-1} \Rightarrow \underline{p}) = (\exists x) (\neg \underline{s} \Rightarrow \underline{p}) \quad (39a)$$

(For completeness sake, we can write (39a) in terms of reverse implications, as

$$(\forall x) (\underline{s} \Leftarrow \underline{a}_1), (\forall x) (\underline{a}_1 \Leftarrow \underline{a}_2), \dots, (\forall x) (\underline{a}_{n-2} \Leftarrow \underline{a}_{n-1}) \\ (\exists x) (\underline{a}_{n-1} \Leftarrow \underline{p}) = (\exists x) (\underline{s} \Leftarrow \underline{p}) \quad (39b)$$

which is seen to be identical in structure with (37a), but with reverse implications (\Leftarrow), instead of implications (\Rightarrow). Just as (37a) has an \exists only for the first implication, (39b) also has an \exists only for the first reverse implication, going from right to left, and \forall 's for all the rest.)

The two cases (α) and (β) (given in (37a) and (39)) exhaust the possibilities under (ii) having some \exists 's present

in the sequence. Under (i), there is only one possibility, which takes the form (36a). Thus, totally, there are only three patterns (that are independent) which have valid conclusions, for a sequence of n QPL implications in statements of Type-1.

References

1. Church, A. (1966) Article on "Logic" in Encyclopaedia Britannica, Vol 14, p,216, Chicago.
2. DeLong, H. (1971) "A Profile of Mathematical Logic", Addison-Wesley, Reading.
3. Ramachandran, G.N. (1979) "Syad-nyaya-system logic — Essential ideas connected with Propositional Calculus", Matphil Reports No. 1, pp 1-38.
4. Ramachandran, G.N. (1980) "Computerization of Logic" Golden Jubilee Commemoration Volume, Natl. Acad. Sci. India, pp 1-42.
5. Ramachandran, G.N. (1982) "Syad Nyaya System (SNS) — A new formulation of sentential logic and its isomorphism with Boolean algebra of genus 2" Current Science, 51, 625-636.
6. Ramachandran, G.N. (1983) (a,b) "Vector-matrix representation of Boolean algebras and application to extended predicate logic (EPL). Part I, 292-302; Part II, 335-341, Current Science, 52.
7. Ramachandran, G.N. (1983c) "Boolean algebra applied to statements of Type-1 in quantified predicate logic (QPL) and its consequences for categorical syllogisms of traditional Logic", Matphil Reports No. 33, pp 1 to 79.
8. Ramachandran, G.N., Johnson, R.E.C., and Thanaraj, T.A. (1980) "Electronic Syād-Nyāya-Yantra (ESNY-2) — Analog computer for sentential logic", Matphil Reports No. 13, pp 1-90.

9. Ramachandran, G.N., and Thanaraj, T.A.(1980a) "Outline of the Fortran Program NYAYA1", Matphil Reports No. 6, pp 1-32.
 10. Ramachandran, G.N., and Thanaraj, T.A. (1980b) "Fortran Program for Sentential Logic", Matphil Reports No. 12, pp 1-61.
 11. Ramachandran, G.N., and Thanaraj, T.A. (1981) "Matrix algebra for sentential logic and the Fortran program MATLOG", Matphil Reports No. 18, pp 1-64.
 12. Ramachandran, G.N., Ramachandran, H., and Thanaraj, T.A. (1981) "Pocket Calculator program TINY1 for sentential logic", Matphil Reports No. 16, pp 1-43.
-

Vector-Matrix Representation of Boolean Algebra and Application
to Extended Predicate Logic (EPL)

Part IV - General Theory for Statements of Type-1 in QPL

G.N. Ramachandran
Gita, 5, 10A Main Road
Mallewaram West
Bangalore 560 055

*Matphil Reports No. 35A, April 1984
(Draft II - for private Circulations)

Vector-Matrix Representation of Boolean Algebras and Application
to Extended Predicate Logic (EPL)

Part IV — General Theory for Statements of Type-1 in QPL*

G.N. Ramachandran
Gita, 5, 10A Main Road
Malleswaram West
Bangalore 560055

*Matphil Reports No. 35, April 1984
(Draft II — for private circulation)

PREFACE AND SUMMARY

This report deals with the generalisation of the theory in Part III dealing with categorical statements in quantified predicate logic (QPL), extended to all statements of Type-1, having the same quantifier $\forall(x)$. The method adopted is to extend the logical graph of an SNS argument containing relations of the type $\underline{a} \underset{\sim}{\approx} \underline{b}$ and $\underline{a} \underset{\sim}{\approx} \underline{b} = \underline{c}$ to the case of an argument containing Type-1 relations in QPL of the type $\forall(x) (\underline{a}(x) \underset{\sim}{\approx} \underline{b}(x))$ and $\forall(x) (\underline{a}(x) \underset{\sim}{\approx} \underline{b}(x) = \underline{c}(x))$. The theory of SNS arguments is briefly dealt with in Section 2, after which the elementary unary and binary statements containing matrix operators $\underset{\sim}{\approx}$ of the above type in QFL Type-1 arguments are discussed in Section 3, along with binary Boolean relations of the type $\underline{\underline{a}}_1 \cup \underline{\underline{a}}_2 = \underline{\underline{a}}$ and $\underline{\underline{a}}_1 \vee \underline{\underline{a}}_2 = \underline{\underline{a}}$.

This is then followed by detailed discussion of the structure of the logical graph of Type-1 argument and how various preliminary alterations of this structure make it possible to convert the

given graph into a "reduced" graph in which every step can be

written in an algorithmic manner with the following structure:

input QFL state(s), QPL Type-1 relation, leading to QFL state

of output, of the particular elementary relation, forming part

of the reduced graph. After discussing the various manipulations

and steps involved in this. a summary of the implementation

procedure is discussed in Section 5 along with a proof that a

reduced graph of the above type is always implementable for

any finite QPL Type-1 logical graph, (* see addendum).

The paper in effect proves the result that QFL logic of

Type-1 has no incompleteness or indefiniteness, associated with

it, except that the QFL state of the output may be also one of

the four non-definite states of extended predicate logic; namely

$\Delta = (1 \ 1 \ 1)$ = indefinite, $\emptyset = (0 \ 0 \ 0)$ = contradiction,

$\Sigma = (0 \ 1 \ 0)$ = some, but not all and not none, $\Theta = (1 \ 0 \ 1) =$

all or none. Incidentally, in Section 2 it is shown that a

.iii.

circular SIS argument, which leads to paradoxes, can always be converted into a graph without circularity, leading to some one of the SIS states being a possible input for the argument. The generalization of this to QPL Type-1 circular arguments is employed in the theory of Section 4.

This particular report is Draft-II of a paper being prepared with this title for publication. For the present, it is being xeroxed for private circulation and for opinion of qualified persons. It is not claimed that there are no errors or omissions in this particular draft.

Addendum

(The proof for the implementability of any finite QPL Type-1 logical graph is extendable to the general case of any finite EPL logical graph, containing both Type-1 and Type-2 statements. This follows from the fact that the graph structure of SNS, QPL Type-1, and general EPL graphs, are isomorphic and can be treated for proof of implementability in exactly the same way. This is proposed to be shown in Part V.)

CONTENTS

	Page No.
1. Introduction	1
2. Logical graph of an SIS argument	
(a) Elementary statements	5
(b) Structure of a general logical graph in SIS logic	9b
(c) SNS logical graphs with circuits	14
(d) SNS logical graph having a directed circuit	18
(i) Condensing of n unary equations in sequence	18
(ii) Cyclic equations	19
(iii) Feedback line	22
(iv) Single-term cyclic equations	24
(e) Algebraic implementation of a general SIS graph	31
3. Logical graph of a QPL argument composed to Type-1 statements.	
(a) Elementary statements of this type	34
(b) Explanation of the new connective operator "into"	37
(c) Justification of Fig.6 for all elementary connective relations in QPL	43
(i) Binary relations for general Z	43
(ii) Unary relations for all matrix connectives	46
(iii) Binary reverse relations for matrix connectives	50
4. Structure and implementation of a general logical graph for Type-1 statements.	52
(a) Preliminary steps for binary reverse connectives	53
(b) Preliminary steps related to \tilde{A} -type unary connectives	56

4. (c) Consistency check for multiple inputs to a term	58
(d) Occurrence of the state Σ in a simple two-step argument containing the vidya operator	62
(e) Incorporation of EPL states in Type-1 arguments	66
(i) Implementation of Δ as input	66
(ii) Implementation of Σ and Θ as inputs	67
(iii) Implementation of \emptyset as input	69
(f) Preliminary steps for unary chains	70
(i) Combination of two Type-1 implications	71
(ii) Inclusion of Ξ -type unary connectives	74
(iii) Associativity of n unary connective relations in a unary path	75
(iv) Implementation of a unary bouquet	76
5. Summary of the implementation procedure for a Type-1 argument	81
(a) Proof that a reduced graph consisting of Type-1 statements is implementable	91
(b) Processing of the argument through the reduced graph.	94
(c) Conclusion	96
List of figures	98
List of tables	100
List of references	100

Vector-Matrix Representation of Boolean Algebras and Application
to Extended Predicate Logic (EPL)

Part IV - General Theory for Statements of Type-1 in QPL

1. Introduction

[1]

In Part III we considered the algebraic theory for combining two categorical statements in QPL to obtain the resultant, which

is also a categorical statement. This was then extended to the

resultant of n categorical statements of type $(\forall_j x)(a_j(x) \Rightarrow$

$a_{j+1}(x))$; By having a_j and a_{j+1} to be affirmative or negative,

the four possible SNS connective operators \perp , $\perp\!\!\!\perp$, $\perp\!\!\!\perp$ and $\perp\!\!\!\perp\!\!\!\perp$ (\exists_j)

are all taken into account. For the quantifier \forall_j , only two

possibilities in QPL need be considered, namely \forall and \exists .

More generally, a statement of Type-1 has the form

$(\forall x)(s(x))$, where $\forall = \forall$ or \exists and $s(x)$ is
an SNS statement (1)

If $s(x) \exists (a(x) \Rightarrow b(x))$, then we get the restricted example

of categorical statements. More generally, however, $s(x)$ can have

various other forms and the only condition we put is that it must be a logical statement composed of SNS terms and connectives, which can take up the four possible truth values T, F, D, X of SNS logic. The theory of SNS statements and sentences has been discussed in Ref [1] and it can be shown that any general SNS argument can be written in a form of a logical graph in which the elementary statements are either unary or binary. These two examples of elementary statements in SNS can be represented symbolically as follows:

Unary $\underline{s}(x) \equiv (\underline{a}(x) \underset{\sim}{\underline{\wedge}} \underline{b}(x))$,

where \sim is one of the ten possible SNS operators, \underline{E} , \underline{N} , \underline{A} , \underline{O} , \underline{I} , \underline{J} , \underline{A}^c , \underline{O}^c , \underline{I}^c , \underline{J}^c (see below for limitations on these). (See Ref [1] (2) and Section 3(c)(ii))

Binary Forward

$\underline{s}(x) \equiv (\underline{a}(x) \sim \underline{b}(x) = \underline{c}(x))$,

where \sim is one of the above ten matrix connectives, or one of the three Boolean connectives \underline{U} , \underline{V} , \underline{W} (see Ref [1] and Section 3(a), below) (3)

Binary Reverse

$$\underline{s}(x) \equiv (\underline{c}(x) \underset{\zeta}{\sim} \underline{a}(x) = \underline{b}(x)),$$

in which ζ can be only one of the above ten matrix connectives! (See Fig. 1 and Section 3(c)(ii)).

(4a)

According to the nature of $\underline{c}(x)$, the binary reverse relation

(4a) becomes the following, using forward relations:

$$\underline{c} = T \mapsto \underline{a} \underset{\zeta}{\sim} \underline{b} ; \quad \underline{c} = F \mapsto \underline{a} \underset{\zeta^c}{\sim} \underline{b} \quad (4b, 4c)$$

$$\underline{c} = D \mapsto \underline{a} \underset{\zeta}{\sim} \underline{b} ; \quad \underline{c} = X \mapsto \underline{a} \underset{\zeta}{\sim} \underline{b} \quad (4d, 4e)$$

More generally, $\underline{s}(x)$ in Eq.(1) can be any general SNS statement

connecting any number of terms via any number of unary and/or

binary logical SNS relations. In all these cases, the final

output of the SNS argument has one of the four SNS states when

each input term contained in the argument is specified to have one

of these four truth values. This general theorem is the result

of the theory of SNS logic (see Ref [b] ^{see}; and also Section 2.2)

below for an outline of the proof using our Boolean algebraic representation.)

In this article, we shall generalize the algebra for any general SNS argument to the case when the quantifiers (\forall/x) is appended before the bracket containing the SNS sentence, so that each term in the argument is a QPL statement for the variable x .

$\alpha(x) \equiv (\forall x)(\underline{a(x)})$
The inputs are QPL terms containing both quantifier state as well

as an SNS truth value and we ask for both the SNS truth value,

as well as the quantifier state, for the final output(s), given

the corresponding quantifies for each of the input terms contained in the full set of statements in the QPL argument. As a preliminary

to this, we shall give a fairly full, but condensed, account of

the way in which any general SNS argument is worked out in the

forward direction. The terms and connectives of any such argument

can be represented by a logical graph in which both the terms

as well as the connective operators are nodes, which are connected

by directed edges indicating the direction in which the logic of

the argument flows. This is discussed in the next section for an

SNS argument and the general way in which the outputs of such a graph are worked out is indicated therein. Certain recondite points such as the reversal of the direction of flow of the argument when the "impossible" state X arises are only briefly commented upon. Similarly, the case of circular arguments, in which the logical graph has a closed circuit of edges are also only briefly discussed. (The difficulties with circular arguments arises only when the "doubtful" state D or the "impossible" state X arises as a consequence.) We shall comment on these at the appropriate place and reserve a full discussion of such pathological situations for a separate report.

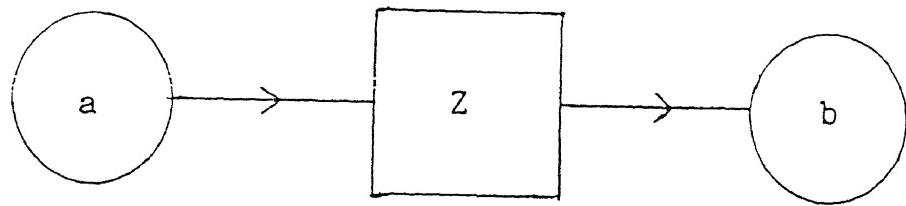
2. Logical graph of an SNS argument

(a) Elementary statements

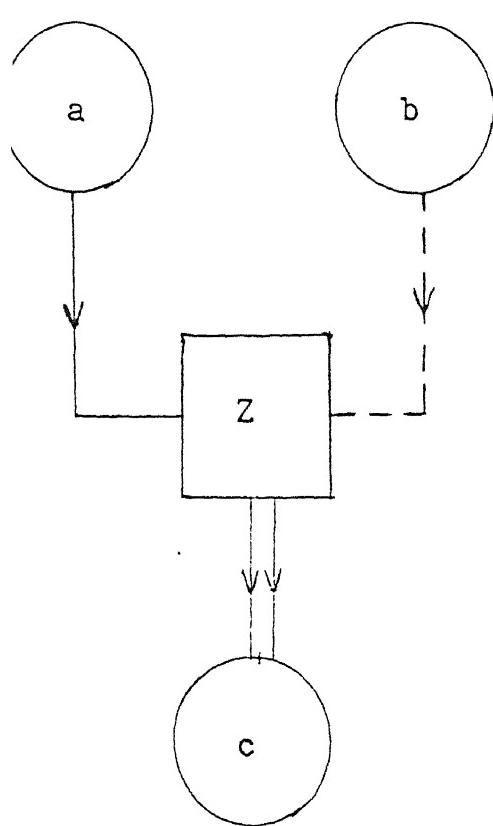
The logical graphs corresponding to the SNS parts of the elementary QPL statements in Eqs. (2), (3) and (4) are shown in Fig. 1, and the corresponding equations are given below in

Fig. 1 Elementary statements of SNS logic.

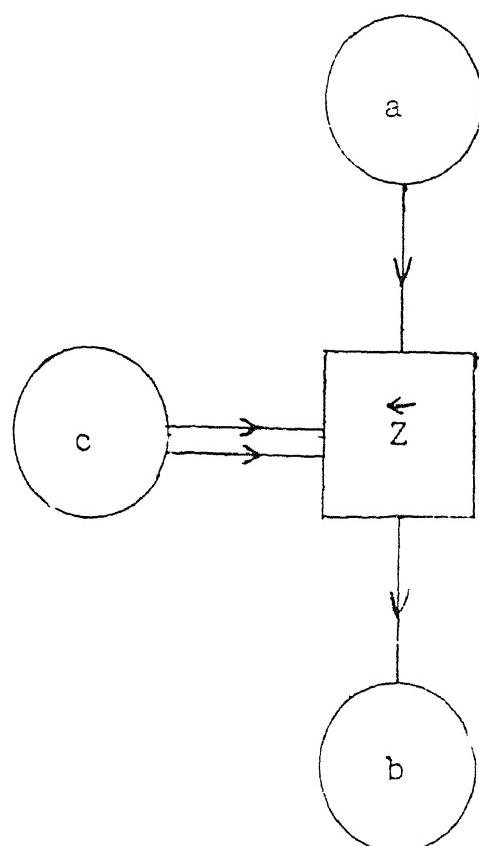
.6.



(a). Unary relation as in Eqs (2) and (5)



(b). Binary forward relation
as in Eqs (3) and (6)



(c). Binary reverse relation
as in Eqs. (4) and (7)

FIG. 1. Elementary statements of SNS logic

.7.

Eqs. (5), (6) and (7):

Unary (SNS) relation

$\underline{\underline{a}} \underset{\sim}{Z} = \underline{\underline{b}}$; Given the input term $\underline{\underline{a}'}$, and the nature of the connective $\sim Z$, to find the output term $\underline{\underline{b}'}$ (5a, b)

The matrix equation for this is

$$\langle a' | Z | = \langle b' | \quad (5c)$$

whose expanded form is

$$\sum_{\lambda=\alpha, \beta} a'_\lambda Z_{\lambda\lambda} = b'_\alpha; \quad \sum_{\lambda=\alpha, \beta} a'_\lambda Z_{\lambda\beta} = b'_\beta \quad (5d)$$

Binary forward (SNS) relation:

$\underline{\underline{a}} \underset{\sim}{Z} \underline{\underline{b}} = \underline{\underline{c}}$; Given the input terms $\underline{\underline{a}'}$ and $\underline{\underline{b}'}$, and the connective $\sim Z$, to find $\underline{\underline{c}'}$ (6a, b)

For matrix operators, the equation is

$$\langle a' | Z | b' \rangle = c'_\alpha, \quad \langle a' | Z^c | b' \rangle = c'_\beta \quad (6c, d)$$

where

$$\langle a' | Z | b' \rangle = \sum_{\lambda} \sum_{\mu} a'_\lambda Z_{\lambda\mu} b'_\mu \quad (6e)$$

For non-matrix binary operators (see Ref.[1]), the equations

are:

$$\underline{\underline{z}} = \underline{\underline{u}} : a'_\alpha \oplus b'_\alpha = c'_\alpha ; a'_\beta \oplus b'_\beta = c'_\beta \quad (6f)$$

$$\underline{\underline{z}} = \underline{\underline{v}} : a'_\alpha \otimes b'_\alpha = c'_\alpha ; a'_\beta \otimes b'_\beta = c'_\beta \quad (6g)$$

Binary reverse(SNS) relation:

$$\underline{\underline{c}} \xleftarrow{\quad} \underline{\underline{a}} = \underline{\underline{b}} ; \text{ Given } \underline{\underline{c}}', \underline{\underline{a}}', \text{ to find } \underline{\underline{b}}' \quad (7a, b)$$

For the four truth values T, F, D, X, respectively, of $\underline{\underline{c}}'$, we get the unary relations, as in (4a-d), and the matrix equations for these are of the same form as in (5c, d). (7c)

It is to be noted that any unary forward relation is expressible as a binary relation, ^{with} when $\underline{\underline{c}}' = T$ or F . Hence, the introduction of the unary relation in our Boolean-algebraic representation of logic is consistent with the conventional treatments of propositional

calculus; but the consequences of including also the SIS states

D and X are covered by our matrix algebraic representation as

given above. In particular, $\underline{\underline{a}} \underset{\sim}{\underline{\underline{Z}}} = \underline{\underline{b}}$ is equivalent to $(c_T \overset{\leftarrow}{\underset{\sim}{\underline{\underline{Z}}}} \underline{\underline{a}}) =$

As will be discussed in particular in connection with QPL statements of Type-1 (see Section 3(c)(ii)), the ten common matrix operators $\underline{\underline{Z}}$ can be classified into three types — namely the A -type ($\underline{\underline{A}}, \underline{\underline{Q}}^C, \underline{\underline{L}}^C, \underline{\underline{J}}^C$), the I -type ($\underline{\underline{Q}}, \underline{\underline{A}}^C, \underline{\underline{L}}, \underline{\underline{J}}$), and the E -type ($\underline{\underline{E}}, \underline{\underline{E}}^C \equiv \underline{\underline{I}}$). Of these, the I -type and E -type operators never give a unary output which is X for $\underline{\underline{a}} \underset{\sim}{\underline{\underline{Z}}} = \underline{\underline{b}}$, for $\underline{\underline{a}} = T$ or F , while the A -type operators can do so. For example, $\underline{\underline{a}}' = \underline{\underline{F}} = (0 \ 1)$ input to $\underline{\underline{a}} \underset{\sim}{\underline{\underline{A}}} = \underline{\underline{b}}$, leads to

$$\underline{\underline{b}}' = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 \end{pmatrix} = X \quad (8)$$

If the structure of the unary relation, as in Fig. 1(a) is adopted for the A -type operators also, then the contradictory state X

will be propagated through the graph. To avoid this, in the analogous unary relation for Type-1 statements, as shown in

Fig. 6a, a new construction called "modified \tilde{A} -construction"

is adopted by which the $\underset{\text{actual}}{\text{input}}$ $(0 \quad 1) = a'_1 T$, and the necessary input $(1 \quad 0) = a'_2 T$ for \tilde{A} , are compared by the consistency

"vidya" operator $\underset{\tilde{A}}{\approx}$ and this leads to an output which may be

X or otherwise. On the other hand, the output from $\underline{a} \underset{\tilde{A}}{\approx} \underline{b}$

goes as $\underline{b}' = T = (1 \quad 0)$ into the graph for further processing.

(As this great convenience of the modified \tilde{A} -construction

became very clear only for Type-1 statements, reference is here

made to Fig. 6b and its explanation in Section 5(c)(ii) and

it is not explained here. However, it may be mentioned that the

net result is that for all operators of the type shown in Fig. 1,

by making suitable \tilde{A} -constructions and \tilde{V} -constructions (see

later in this section), it is possible to relegate all contradictory

states X to outputs and the graph can be analysed and implemented

throughout. The proof of this is given later in this section.)

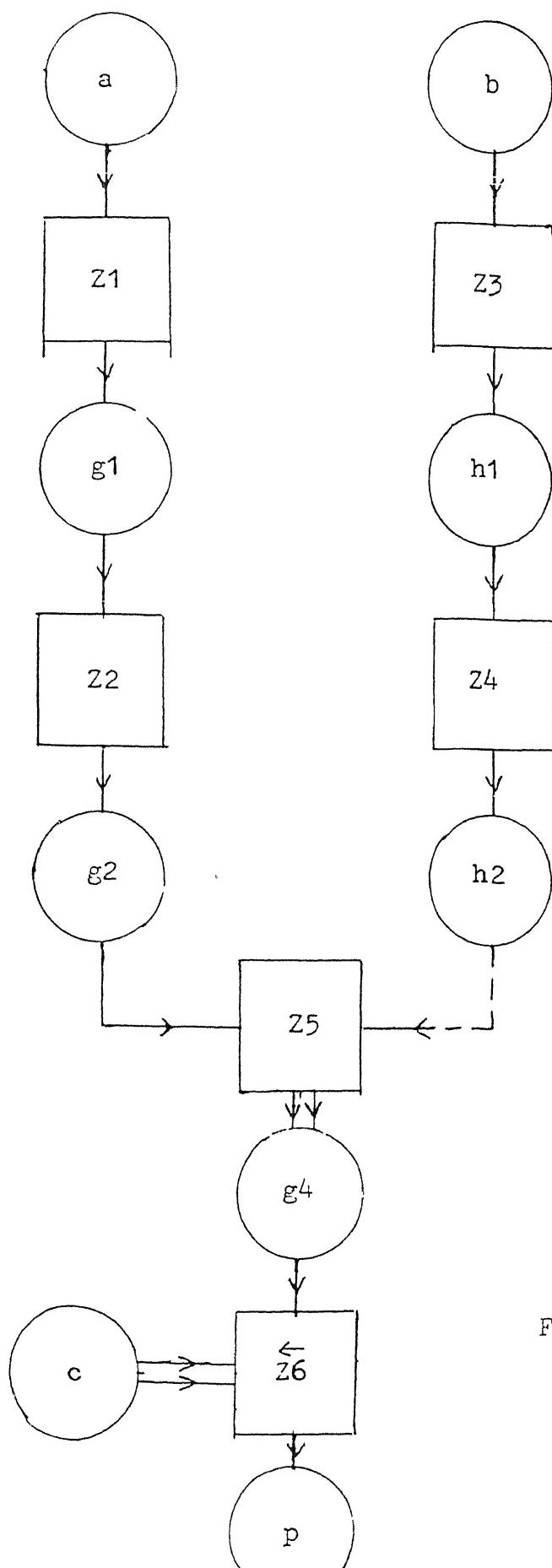
(b) Structure of a general logical graph in SLS logic.

We shall not make the treatment in this section completely general. Instead, we shall consider two or three typical logical graphs of this type and indicate how the steps from the inputs to the final output are worked out in each case. Thereafter, we shall try to indicate how the whole process can be generalized and worked out for any logical graph.

To start with, we shall consider a graph which has no circuits and has therefore the structure of a "tree" (Fig.2). The nature of the six connective operators $\underline{Z_1}$ to $\underline{Z_6}$ is also indicated within Fig. 2. It is to be noted that the edges leading to \underline{c} in

Fig.2. Logical graph having a tree structure containing unary connectives and binary forward and binary reverse connectives.

$\underline{a} \underline{\sim} \underline{b} = \underline{c}$, and starting from \underline{c} in $\underline{c} \overset{\leftarrow}{\underline{\sim}} \underline{a} = \underline{b}$, are marked by double lines (see Fig. 1), and the input from \underline{b} to $\underline{\sim}$ is distinguished from that from \underline{a} , in $\underline{a} \underline{\sim} \underline{b}$, by a dotted line for the former



\tilde{z}_1	=	\overline{I}
\tilde{z}_2	=	$\overline{E}^C(N)$
\tilde{z}_3	=	$\overline{IN}_{\sim}(A^C)$
\tilde{z}_4	=	$\overline{NI}_{\sim}(O)$
\tilde{z}_5	=	\overline{A}
\tilde{z}_6	=	\overline{I}

Inputs: $\underline{a}, \underline{b}, \underline{c}$

Output: \underline{p}

Fig. 2. Logical graph having a tree structure containing unary connectives and binary forward and reverse connectives.

The equations connecting the various terms that are contained in this graph are given below.

$$\underline{\underline{a}} \underline{\underline{z}_1} = \underline{\underline{g}_1} ; \quad \underline{\underline{g}_1} \underline{\underline{z}_2} = \underline{\underline{g}_2} \quad (8a, 8b)$$

$$\underline{\underline{b}} \underline{\underline{z}_3} = \underline{\underline{h}_1} ; \quad \underline{\underline{h}_1} \underline{\underline{z}_4} = \underline{\underline{h}_2} \quad (8c, 8d)$$

$$\underline{\underline{g}_2} \underline{\underline{z}_5} \underline{\underline{h}_2} = \underline{\underline{g}_4} ; \quad \underline{\underline{c}} \stackrel{\leftarrow}{\underline{\underline{z}_6}} \underline{\underline{g}_4} = \underline{\underline{p}} \quad (8e, 8f)$$

It is to be noted that only matrix-type connectives are used in this graph. Further, Eq. (8f) effectively yields the equation $\underline{\underline{c}} \underline{\underline{z}_7} = \underline{\underline{p}}$, in which $\underline{\underline{z}_7} = \underline{\underline{I}}$ or $\underline{\underline{AN}}$, according as $\underline{\underline{c}} = T$ or F .

Using Eqs (5), (6) and (7) and the 2X2 Boolean matrix representations of the various connectives (as given in Table 2 of Ref.[2]), we can work out the results shown in Table 1 below for the SNS truth values of the output $\underline{\underline{p}}$, and of the intermediate terms $\underline{\underline{g}_1}$, $\underline{\underline{g}_2}$, $\underline{\underline{h}_1}$, $\underline{\underline{h}_2}$, $\underline{\underline{g}_4}$, corresponding to the eight possibilities, obtained by putting each of $\underline{\underline{a}}$, $\underline{\underline{b}}$, $\underline{\underline{c}}$ equal to T and F.

Table 1. SNS truth values of the terms in Fig. 1 for the different T and F inputs for $\underline{\underline{a}}$, $\underline{\underline{b}}$, $\underline{\underline{c}}$.

Table 1. SNS truth values of the terms in Fig.1 for the different T and F inputs for \underline{a} , \underline{b} , \underline{c} .

Sl. No.	Inputs			Intermediate output-inputs					$\underline{\underline{Z}}\underline{\underline{7}}$	Output \underline{p}
	$\underline{\underline{a}}$	$\underline{\underline{b}}$	$\underline{\underline{c}}$	$\underline{\underline{g}}\underline{1}$	$\underline{\underline{g}}\underline{2}$	$\underline{\underline{h}}\underline{1}$	$\underline{\underline{h}}\underline{2}$	$\underline{\underline{g}}\underline{4}$		
1	T	T	T	T	F	F	T	F	I	T
2	T	T	F	T	F	F	T	F	AN	X
3	T	F	T	T	F	D	D	F	I	T
4	T	F	F	T	F	D	D	F	AN	X
5	F	T	T	D	D	F	T	D	I	D
6	F	T	F	D	D	F	T	D	AN	F
7	F	F	T	D	D	D	D	D	I	D
8	F	F	F	D	D	D	D	D	AN	F

It will be noticed that, in four cases, \underline{p} has the definite state T or F; and in two cases it has the doubtful state D. What is interesting is that, in two cases, namely Sl. No. 2; $\underline{g} = T$, $\underline{b} = T$, $\underline{c} = F$, and Sl. No. 4: $\underline{g} = T$, $\underline{b} = F$, $\underline{c} = F$, the output \underline{p} has the impossible state X, and ^{However,} this impossible state (indicating a contradiction) occurs only at the last step involving $\underline{\tilde{z}}\tilde{6}$ and \underline{c} .

We can therefore conclude that, in these two cases, if all the earlier steps, as well as the inputs \underline{g} and \underline{b} , are perfectly valid, then \underline{c} can only be T and not F (leading to Sl. Nos 1 and 3).

Vice versa, if \underline{c} is known to be definitely F, then \underline{g} can only be F and not T (as in Sl. Nos 6 and 7). It is readily verified that, in all these four cases thus corrected, the output \underline{p} is either T or F, and not X. (We shall not examine further the theory of reversing the argument from a contradictory state (X) for the output, leading to "conditional" consequences containing the exclusive or, like the above — "Either \underline{c} is to be changed to T or \underline{g} must be changed to F" — which occurs in this argument. This is reserved for a separate communication.)

Table 2. List of intermediate terms and output p and q
for the logical graph shown in Fig. 3

Sl. No.	Inputs		Intermediate output-inputs				Outputs	
	$\underline{\underline{a}}$	$\underline{\underline{b}}$	$\underline{\underline{g}}^1$	$\underline{\underline{p}}^1$	$\underline{\underline{h}}^1$	$\underline{\underline{p}}^2$	$\underline{\underline{p}}$	$\underline{\underline{q}}$
1	T	T	T	D	F	F	F	T
2	T	F	T	X	F	F	X	T
3	F	T	D	D	D	D	D	D
4	F	F	D	F	D	D	F	D

(c) SNS logical graphs with circuits

We shall first consider an example of a graph with a closed circuit of edges (Fig. 3), in which the circuit does not have throughout the same direction for the flow of the argument.

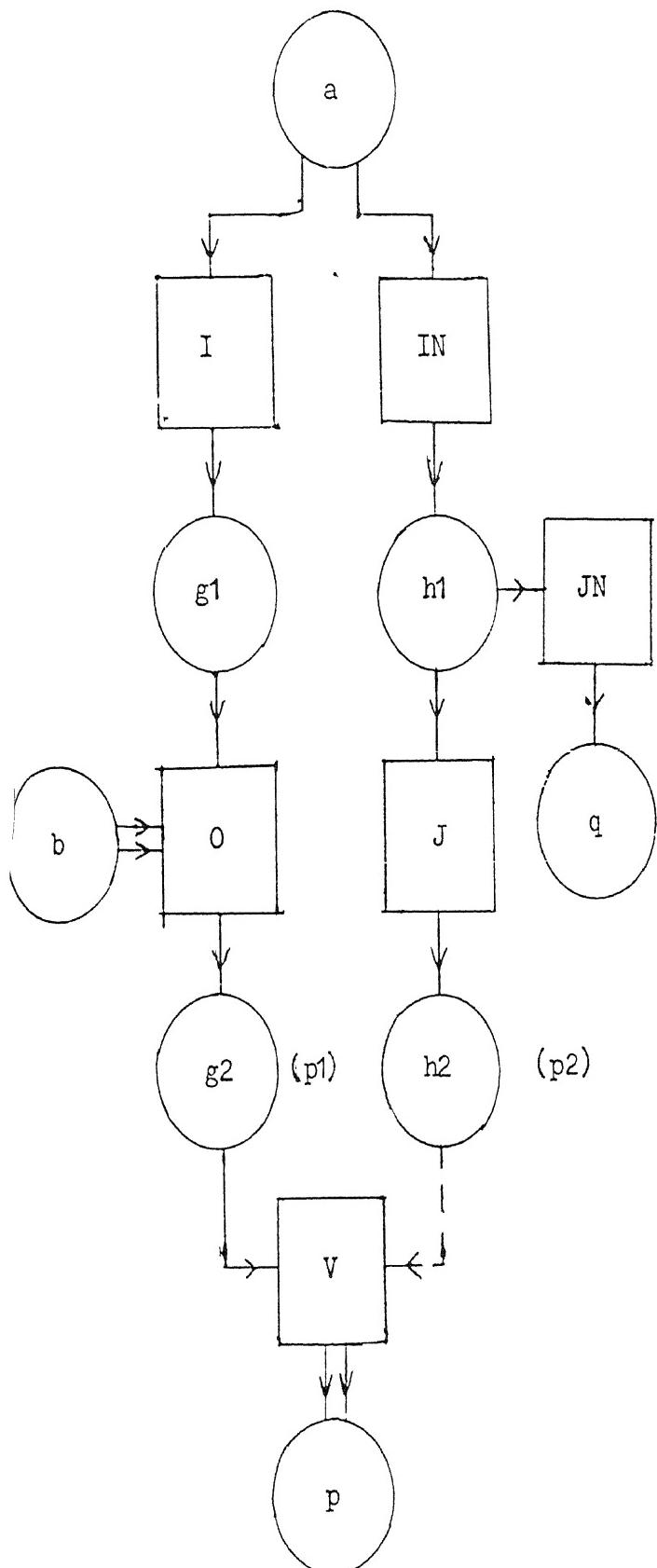
The nature of the connectives is indicated in the graph itself.

The results for the various intermediate terms and for the outputs \underline{p} and \underline{q} corresponding to possible truth values T and F for the inputs \underline{a} and \underline{b} are given in Table 2.

Table 2. List of intermediate terms and outputs \underline{p} and \underline{q} for the logical graph shown in Fig. 3

Fig. 3. A simple logical graph having a closed circuit of edges, in which the outputs from two different paths are combined by the Boolean connective $\underline{\vee}$

It is to be noted that \underline{p}_1 and \underline{p}_2 are the truth values giving information regarding the state of \underline{p} coming via two different paths, which are combined by the vidya operator $\underline{\vee}$ to give the values of p_α and p_β that is common to them, so as to give $\underline{p} = (p_\alpha \ p_\beta)$ (see Eq (6.2)).



$$\underline{\underline{a}} \underline{\underline{I}} = \underline{\underline{g}} \underline{\underline{1}} \quad (\text{Step 1})$$

$$\underline{\underline{b}} \overset{\leftarrow}{\underline{\underline{O}}} \underline{\underline{g}} \underline{\underline{1}} = \underline{\underline{g}} \underline{\underline{2}} = \underline{\underline{p}} \underline{\underline{1}} \quad (\text{" 2})$$

$$\underline{\underline{a}} \underline{\underline{I}} \underline{\underline{N}} = \underline{\underline{h}} \underline{\underline{1}} \quad (\text{" 3})$$

$$\underline{\underline{h}} \underline{\underline{1}} \underline{\underline{J}} = \underline{\underline{h}} \underline{\underline{2}} = \underline{\underline{p}} \underline{\underline{2}} \quad (\text{" 4})$$

$$\underline{\underline{p}} \underline{\underline{1}} \vee \underline{\underline{p}} \underline{\underline{2}} = \underline{\underline{p}} \quad (\text{" 5})$$

$$\underline{\underline{h}} \underline{\underline{1}} \underline{\underline{JN}} = \underline{\underline{q}} \quad (\text{" 6})$$

Fig.3. A simple logical graph having a closed circuit of edges, in which the outputs from two different paths are combined by the Boolean connective $\underline{\underline{\vee}}$.

It will be noticed that no contradiction arises for the

output \underline{q} . On the other hand, $\underline{p} = X$ for the inputs $\underline{a} = T$, $\underline{b} = F$,

but the impossible state X arises for the first time for \underline{p}_1 ,

produced by Step 2 in the graph. Hence, one of the two inputs

to $\overset{\leftarrow}{\Sigma}$, namely \underline{b} and \underline{g}_1 , must be changed. If \underline{b} is made T, then

\underline{g}_2 ($= \underline{p}_1$) is D, for $\underline{a} = T$, and we get the case of Sl.No. 1 for the

inputs, with the X-state being eliminated. On the other hand, if

we know for certain that $\underline{b} = F$, the equation for step 2 becomes

$F \overset{\leftarrow}{\Sigma} \underline{g}_1 = \underline{g}_2$ ($= \underline{p}_1$), or $\underline{g}_1 \overset{\leftarrow}{\Sigma}^C = \underline{g}_2$, which requires that $\underline{g}_1 = F$.

On reversing step 1 in Fig. 3, we have $\underline{g}_1 \underset{\sim}{\Sigma} \text{NIN} = \underline{a}$ and this yields

$\underline{a} = F$ for the input $\underline{g}_1 = F$. Thus, we get a conditional consequence

of the contradiction X for \underline{p}_1 ($= \underline{g}_2$), namely, "Either \underline{b} is to be

changed from F to T, or \underline{a} to be changed from T to F", a consequence

very similar to that of an X arising in Table 1 for Fig. 2.

The other interesting feature in Fig. 3 is the occurrence of the S+S Boolean operator $\underset{\sim}{\Sigma}$, as mentioned above. It removes the

doubt in one of p₁ and p₂ when the other input has a definite state T or F, in Step 5 (as in Sl. No. 1 of Table 2). If, however, one of p₁ and p₂ were T and the other F (as does not happen in Table 2) it would lead to

$$\underline{\underline{p}} = \begin{matrix} T & V \\ \sim & \sim \end{matrix} F = (1 \quad 0) \otimes (0 \quad 1) = (0 \quad 0) \equiv X \quad (9)$$

If so, such a contradiction (X) for the state (truth value) of p would indicate that "Either p₁ has to be changed from T to F, or p₂ has to be changed from F to T" and the consequence in either case worked back in the reverse sense of the argument, to the inputs. An example of this will be shown in the next subsection.

(d) SNS logical graph having a directed circuit

(i) Condensing of n unary equations in sequence

Suppose we have (n-1) statements, all unary, as in Eq (10)

below:

$$\underline{\underline{a}} \underset{\sim}{\sim} \underline{\underline{z}}_1 = \underline{\underline{g}}_1, \quad \underline{\underline{g}}_1 \underset{\sim}{\sim} \underline{\underline{z}}_2 = \underline{\underline{g}}_2, \quad \dots, \quad \underline{\underline{g}}_{(n-2)} \underset{\sim}{\sim} \underline{\underline{z}}_{(n-1)} = \underline{\underline{b}} \quad (10)$$

Then, they can be readily combined to give a single equation

$$\underline{\underline{a}} \underline{\underline{Z}} = \underline{\underline{b}}, \quad |Z| = |z_1| z_2 | \dots | z_{(n-2)} | \quad (11a, b)$$

If we are not interested in the intermediate terms $\underline{\underline{g1}}$ to $\underline{\underline{g(n-2)}}$,

then (11a) can be employed for all calculations pertaining to

a logical graph, of which (10) is a subgraph. The equations

(11a, b) can also be written in the reverse sense, from $\underline{\underline{b}}$ to $\underline{\underline{a}}$,

for which the relevant equation, obtained by reversing the sense

of all the $(n-1)$ equations in (10), is

$$\underline{\underline{b}} \underline{\underline{Z^t}} = \underline{\underline{a}}, \quad |Z^t| = |z_{(n-2)}^t| \dots |z_2^t| z_1^t | \quad (12a, b)$$

The analogues of the condensed equations (11a) and (11b) are

very relevant to QPL statements of type-1, and they will be

discussed in Section below.

(ii) Cyclic equations

We shall consider now a cyclic set of equations in SNS logic,

obtained by augmenting (10) by adding one more equation $\underline{\underline{b}} \underline{\underline{Z_n}} = \underline{\underline{a}}$.

This yields a two-equation circular argument shown in Fig. 4(a):

$$\underline{\underline{a}} \circ \underline{\underline{Z}}' = \underline{\underline{b}}, \quad \underline{\underline{b}} \circ \underline{\underline{Z}}'' = \underline{\underline{a}} \quad (12a, b)$$

with

$$\underline{\underline{Z}}' = \underline{\underline{z}}_1 \underline{\underline{z}}_2 \dots \underline{\underline{z}}_{(n-1)}, \quad \underline{\underline{Z}}'' = \underline{\underline{z}}_n \quad (13a, b)$$

We shall consider a few combinations of $\underline{\underline{Z}}'$, $\underline{\underline{Z}}''$ in (12) to see how the circularity of the argument affects them. If $\underline{\underline{Z}}' = \underline{\underline{Z}}'' = \underline{\underline{I}}$, $a_T \mapsto b_T$, $b_T \mapsto a_T$; but $a_F \mapsto b_D$, $b_D \mapsto a_D$. Hence, there is only one solution possible for the two equations (12a, b) viz. $\underline{\underline{a}} = T$, $\underline{\underline{b}} = T$. All other possibilities lead to the trivial state.

If $\underline{\underline{Z}}' = \underline{\underline{I}}$, $\underline{\underline{Z}}'' = \underline{\underline{IN}}$, and we start with $\underline{\underline{a}} = T$, we obtain $a_T \mapsto b_T$, but $b_T \mapsto a_D$ and on continuing the cycle, $a_D \mapsto b_D$, $b_D \mapsto a_D$. Hence, starting with a_T does not lead to a solution with either T or F for $\underline{\underline{a}}$ and $\underline{\underline{b}}$. On the other hand, if $\underline{\underline{a}} = F$, $a_F \mapsto b_D$, $b_D \mapsto a_D$ and once again $\underline{\underline{a}} = D$, $\underline{\underline{b}} = T$ is the only solution obtained. Hence, the graph of Fig. 4(a) has no solution.

Fig.4. (a) Directed circuit with two unary connectives,

(b) Inclusion of the operator V and removal of the directed circuit. Note the feed-back to $\underline{\underline{z}}$

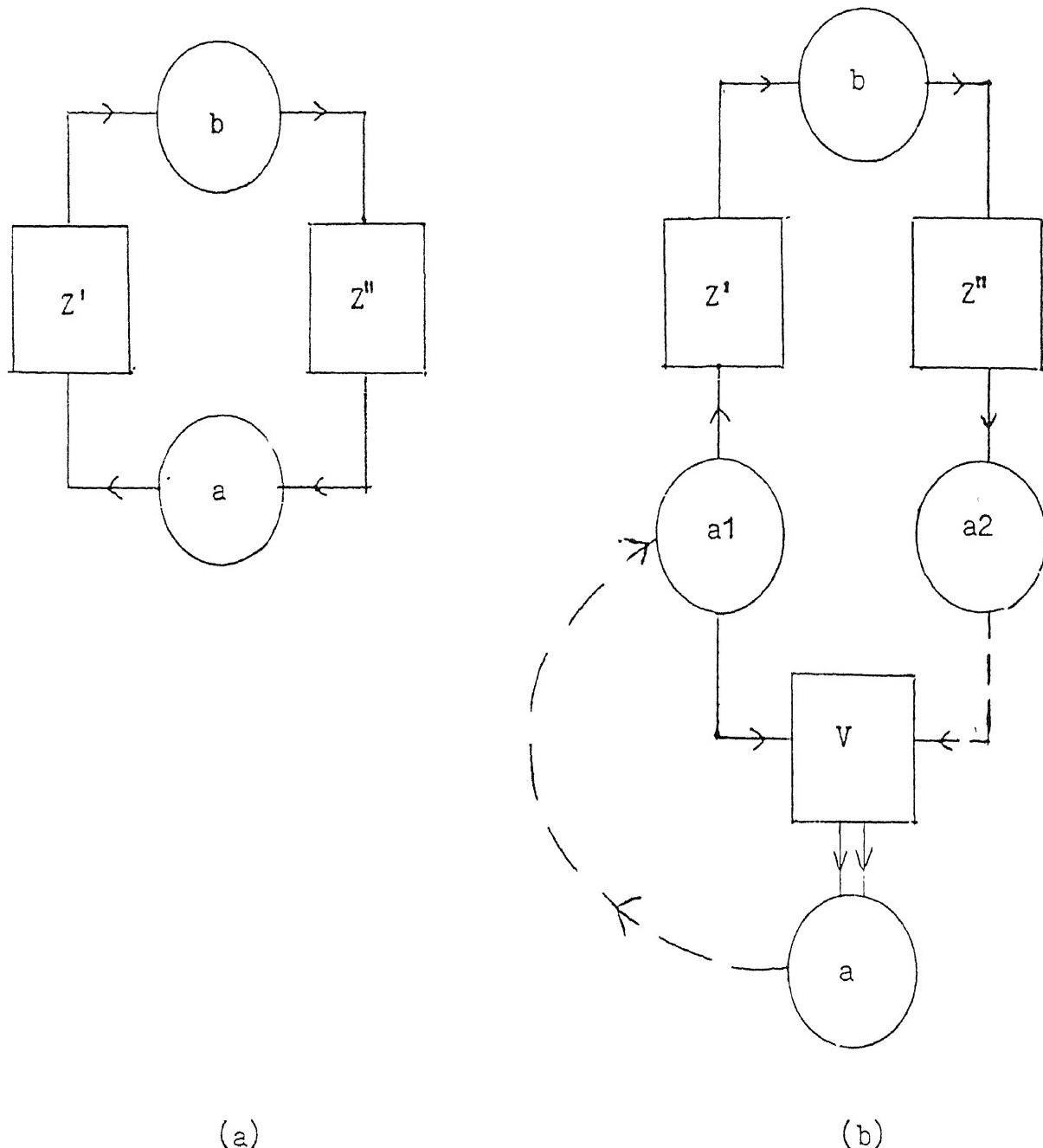


Fig.4. (a) Directed circuit with two unary connectives.
 (b) Inclusion of the operator V and removal of the directed circuit. Note the feed-back line from a to a_1

with the definite state, but only with the indefinite state D,

when $\underline{Z}' = \underline{I}$ and $\underline{Z}'' = \underline{\text{IN}}$. We will now consider another such pathological case, with $\underline{Z}' = \underline{E}$, $\underline{Z}'' = \underline{N}$. Then if $\underline{a} = T$, we get the unending sequence $a_T \mapsto b_T \mapsto a_F \mapsto b_F \mapsto a_T \mapsto \dots$. We get, for both \underline{a} and \underline{b} , the repeating sequence $\dots T, F, T, F, \dots$.

To avoid this, we can use the vidya operator, as in Fig. 4(b). Since we start from \underline{a} and end with \underline{a} in Fig. 4(a), we call the two terms at the beginning and the end of the circuit as $\underline{\underline{a}1}$ and $\underline{\underline{a}2}$ respectively and test for their consistency by introducing the binary \underline{V} between them. When this is done, the graph does not have a closed circuit with the direction of flow of logic being the same throughout. In Fig. 4(b), it is clockwise from $\underline{\underline{a}1}$, via $\underline{Z1}$, \underline{b} , $\underline{Z2}$ and $\underline{\underline{a}2}$, to \underline{V} , but it is counter-clockwise from $\underline{\underline{a}1}$ to \underline{V} . We shall show that no infinite repetition occurs for this.

(iii) Feedback line

Thus, taking $\underline{Z}' = \underline{I}$, $\underline{Z}'' = \underline{I}$ as in the example above, we obtain $\underline{\underline{a}2} = T$ starting from $\underline{\underline{a}1} = T$; and $\underline{\underline{a}1} \underline{V} \underline{\underline{a}2} = T$. The curved

This yields a two-equation circular argument shown in Fig. 4(a):

$$\underline{a} \underline{Z}' = \underline{b}, \quad \underline{b} \underline{Z}'' = \underline{a} \quad (12a, b)$$

with

$$\underline{Z}' = \underline{\underline{z}}_1 \underline{\underline{z}}_2 \dots \underline{\underline{z}}_{(n-1)}, \quad \underline{Z}'' = \underline{\underline{z}}_n \quad (13a, b)$$

We shall consider a few combinations of $\underline{Z}', \underline{Z}''$ in (12) to

see how the circularity of the argument affects them. If $\underline{Z}' = \underline{Z}''$

$= \underline{I}$, $a_T \mapsto b_T$, $b_T \mapsto a_T$; but $a_F \mapsto b_D$, $b_D \mapsto a_D$. Hence,

there is only one solution possible for the two equations (12a, b)

viz. $\underline{a} = T$, $\underline{b} = T$. All other possibilities lead to the doubtful

state.

If $\underline{Z}' = \underline{I}$, $\underline{Z}'' = \underline{IN}$, and we start with $\underline{a} = T$, we obtain

$a_T \mapsto b_T$, but $b_T \mapsto a_D$ and on continuing the cycle, $a_D \mapsto b_D$

$b_D \mapsto a_D$. Hence, starting with a_T does not lead to a solution

with either T or F for \underline{a} and \underline{b} . On the other hand, if $\underline{a} = F$,

$a_F \mapsto b_D$, $b_D \mapsto a_D$ and once again $\underline{a} = D$, $\underline{b} = D$ is the only

solution obtained. Hence, the graph of Fig. 4(a) has no solution

Fig. 4. (a) Directed circuit with two unary connectives.

(b) Inclusion of the operator \underline{V} and removal of the directed circuit. Note the feed-back line from a to a_1

dotted line indicates that this final output $\underline{\underline{T}}$ for $\underline{\underline{a}}$ is to be fed back to $\underline{\underline{a}}_1$, as in a computer cycle. Such a feedback leads to everything being satisfactory and consistent in this example.

If now we take $\underline{\underline{Z}}' = \underline{\underline{I}}$, $\underline{\underline{Z}}'' = \underline{\underline{IN}}$, $\underline{\underline{a}}_1 = \underline{\underline{T}}$ leads to $\underline{\underline{a}}_2 = \underline{\underline{D}}$ and $\underline{\underline{a}} = \underline{\underline{a}}_1 \vee \underline{\underline{a}}_2 = \underline{\underline{T}}$, which is what is to be fed back to $\underline{\underline{a}}_1$. The fed-back value of $\underline{\underline{a}}_1$ is identical with the starting one. Note that, in the main graph, without the dotted feedback line, $\underline{\underline{a}}_1$ is the input, while $\underline{\underline{a}}_2$ is only an intermediate input-output, leading to the output $\underline{\underline{a}}$.

It is therefore to be noted that the feedback line is another elementary component of a logical graph that we must include in addition to the set of three connectives shown in Fig. 1.

Consider now the third case of subsection (ii), namely $\underline{\underline{Z}}' = \underline{\underline{E}}$, $\underline{\underline{Z}}'' = \underline{\underline{N}}$. Then, we obtain the following for Fig. 4(b)

$$(\underline{\underline{a}}_1 = T) \rightarrow (\underline{\underline{a}}_2 = F) \rightarrow \underline{\underline{a}} = X \quad (13a)$$

$$(\underline{\underline{a}}_1 = F) \rightarrow (\underline{\underline{a}}_2 = T) \rightarrow \underline{\underline{a}} = X \quad (13b)$$

Hence, both the inputs T and F for $\underline{a_1}$ lead to contradictions,

so that the set of equations,

$$\underline{a_1} \underset{\text{E}}{\sim} = \underline{b}, \quad \underline{b} \underset{\text{N}}{\sim} = \underline{a_2}, \quad \underline{a_1} \underset{\text{V}}{\sim} \underline{a_2} = \underline{a}$$

$(14a, b, c)$
~~c, d, e~~
~~(14a, b, c)~~

have no solution, with a definite T or F, for the input $\underline{a_1}$.

This means that the first two equations in $(14a, b)$ are

mutually contradictory and the consequence of the state X for \underline{a}

irrespective of the state T or F for $\underline{a_1}$ indicates the following:

"Either E must be changed to $E^c (= N)$ in $(14a)$, or N must be changed to $N^c (= E)$ in $(14b)$ ", — a conditional conclusion very similar to what

we obtained for the logical graphs in Fig.2 and Fig. 3,

when a contradiction X arises.

(iv) Single-term cyclic equations

Eq.(12), containing two unary relations can be condensed still further to give an one-term circular argument, as in (15a), whose graph is shown in Fig.5(a).

$$\underline{a} \underset{\text{Z}}{\sim} = \underline{a}, \quad \underline{Z} = \underline{Z}' \underset{\text{Z''}}{\sim}$$

(15a, b)

Fig.5 (a) Closed directed circuit with a single term and only one connective

(b) Same as (a), but avoiding the directed circuit, but with a feed-back line.

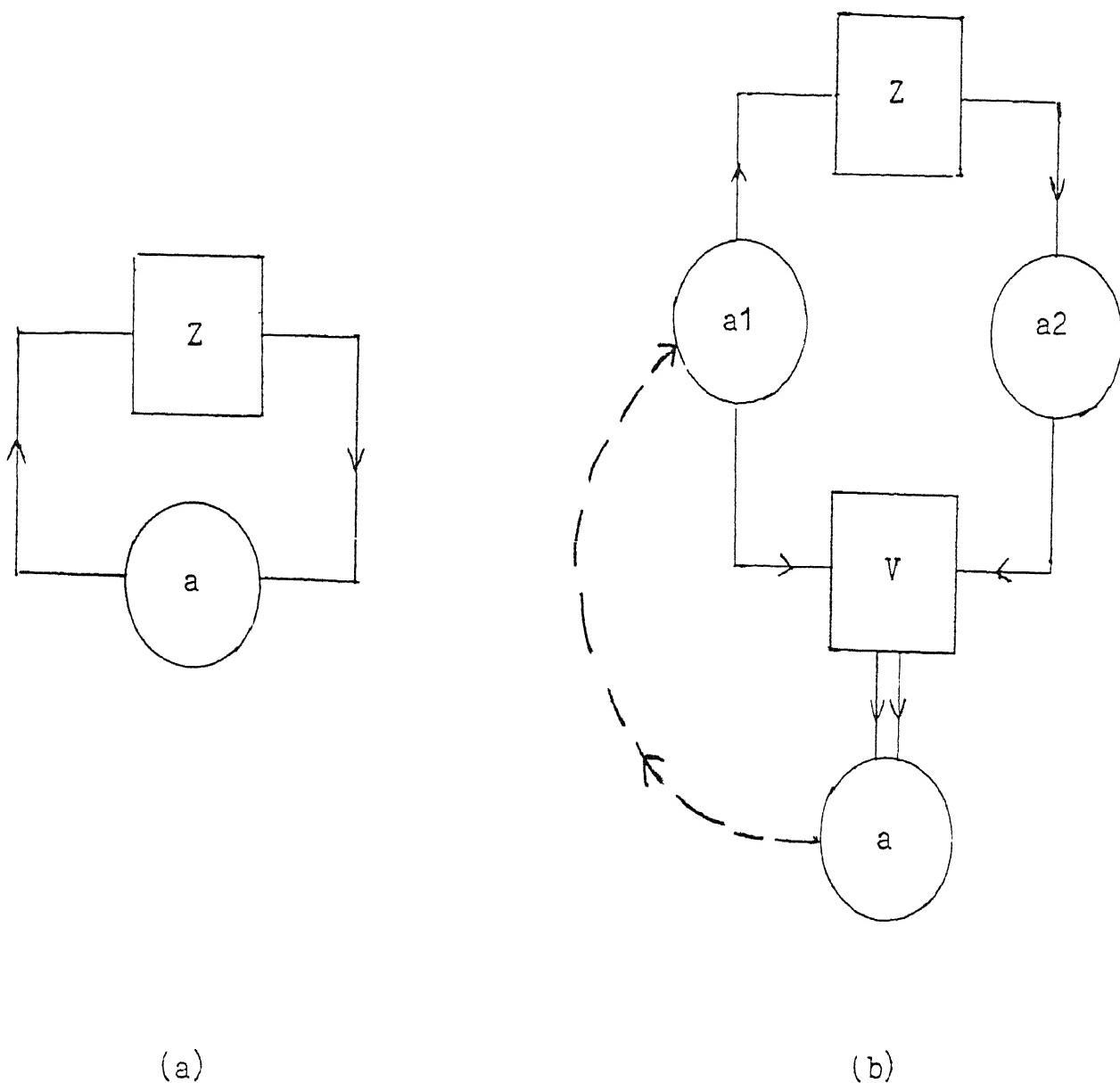


Fig. 5. (a) Closed directed circuit with a single term and only one connective
(b) Same as (a), but avoiding the directed circuit, but with a feed-back line.

We shall consider the behaviour of this for typical cases of \underline{z} being $\underline{\underline{I}}$, $\underline{\underline{IN}}$, $\underline{\underline{E}}$ and $\underline{\underline{N}}$. We denote, for convenience, the input $\underline{\underline{a}}$ in (15a) by $\underline{\underline{a}}_1$ and the output by $\underline{\underline{a}}_2$, and the output is fed as input straightaway for the next cycle. Then, we obtain, for $\underline{\underline{z}} = \underline{\underline{I}}$, the following:

$$\underline{a}_T \mapsto \underline{a}_T \mapsto \underline{a}_T \mapsto \dots \quad (15c)$$

$$\underline{a}_F \mapsto \underline{a}_D \mapsto \underline{a}_D \mapsto \dots \quad (15d)$$

Hence, $\underline{\underline{a}} = T$ is a solution for this case, having a definite truth value.

If now $\underline{\underline{z}} = \underline{\underline{IN}}$,

$$\underline{a}_T \mapsto \underline{a}_F \mapsto \underline{a}_D \mapsto \underline{a}_D \dots \quad (16a)$$

$$\underline{a}_F \mapsto \underline{a}_D \mapsto \underline{a}_D \mapsto \dots \quad (16b)$$

so that, the only stable solution of (15a) for $\underline{\underline{z}} = \underline{\underline{IN}}$ is $\underline{\underline{a}} = D$, which has the doubtful state.

For the case when $\underline{\underline{z}} = \underline{\underline{E}}$,

$$\underline{a}_T \mapsto \underline{a}_T \mapsto \dots ; \quad \underline{a}_F \mapsto \underline{a}_F \mapsto \dots \quad (17a, b)$$

and both $\underline{a} = T$ and $\underline{a} = F$ are possible solutions.

Only for $\underline{Z} = \underline{N}$ do we get a pathological result, as in

(18a, b):

$$a_T \mapsto a_F \mapsto a_T \mapsto \dots ; \quad a_F \mapsto a_T \mapsto a_F \mapsto \dots \quad (18a,b)$$

Thus, we get an endless alternation in the truth value of \underline{a} ,

irrespective of whether we start from T or F.

However, this is not a paradox, but a contradiction, as is shown by converting the graph in Fig. (5a) into that in Fig. 5(b), with the vidya operator checking for contradiction, and including also the feedback line (as in Fig. 4(b)). For a general \underline{Z} , the logical graph in Fig. 5(b) corresponds to the equations:

$$\underline{\underline{a}}_1 \underline{Z} = \underline{\underline{a}}_2 , \quad \underline{\underline{a}}_1 \vee \underline{\underline{a}}_2 = \underline{a} , \quad \underline{a} \mapsto \underline{\underline{a}}_1 \text{ (if } \underline{a} \neq X\text{)} \quad (19a,b,c)$$

The following results for $\underline{Z} = \underline{N}$ are readily worked out:

$$(\underline{\underline{a}}_1 = T) \mapsto (\underline{\underline{a}}_2 = F) \mapsto (\underline{a} = X) \quad (20a)$$

$$(\underline{\underline{a}}_1 = F) \mapsto (\underline{\underline{a}}_2 = T) \mapsto (\underline{a} = X) \quad (20b)$$

Hence the argument contained in the graph shown in Fig. 5(b)

with $\underline{Z} = \underline{N}$ is illogical and self-contradictory, exactly as is
 $\boxed{\text{Eq. (13) for}}$

the case with Fig. 4(b). If we wish to remove this contradiction,

it can only be done by changing the argument itself, i.e. by

changing \underline{Z} to \underline{Z}^C - in this case by changing \underline{N} to $\underline{N}^C (\equiv \underline{E})$. When

we do this, (20a, b) go over to (17a, b) which are perfectly normal.

Another remedy is to admit the doubtful state D as input, when we

obtain (for $\underline{Z} = \underline{N}$ in Fig. 5(b)):

$$(\underline{a}_1 = D) \rightarrow (\underline{a}_2 = D) \rightarrow (\underline{a} = D) \rightarrow (\underline{a}_1 = D) \text{ (consistent)} \quad (21)$$

The equation $\underline{a} \underline{N} = \underline{a}$ is the algebraic representation,

in our notation, of the famous Cretan liar paradox: "What I say is

false", which leads to an endless repetition of the truth values

T, F, T, F, \dots for the statement $\underline{a} \not\models$ "What I say". But, we have

got two solutions (a) $\underline{a} = X$, ^{and} or (b) $\underline{a} = D$, which satisfy this

equation. The truth value X for \underline{a} means that the statement is

"impossible", or "invalid", and there can be no such statement in

logic (propositional calculus). The second possible solution

$\underline{a} = D$ means that "we do not know whether the statement \underline{a} is true

or false", or "the statement \underline{a} may be true, or may be false",

a possible way of expressing its doubtful logical state which is

permissible in SNS logic. Since the state D occurs very frequently

in the analysis of arguments in PC, as in those represented by

the graphs in Figs. 2 and 3, there should be no objection to having

it as the state of the only term in the circular argument $\underline{a} \underset{\sim}{N} = \underline{a}$.

It will be shown below that, if circular arguments in directed

circuits are removed by introducing the vidya operator, as in

Figs 4(b) and 5(b), then any general logical graph in FC can be

converted into a set of elementary logical equations, which can

then be implemented in sequence (see Section 2(1)). In such a case,

the consequences of the inputs having given truth values can

always be worked out and the logical states of the outputs can be

determined by straightforward algorithms. Such algorithms can even

be converted into a computer program. This is discussed in the next subsection.

We shall not describe here the procedure whereby contradictions, when they arise, are used to work backwards into the argument and find out the precise location of the place where the contradiction arises, and the consequent change that should be introduced into truth values, or nature of connectives, in earlier steps in the argument. The general treatment of this is a big project and has not yet been completed. It will be discussed in a later part of this series. Here, we shall only point out, ⁱⁿ the next subsection, the way in which SNS graphs are implemented and worked out, because this is vital for giving a general theory of the implementation of a QPL argument completely consisting of Type-1 statements.

(e) Algebraic implementation of a general SIS graph

Any general SIS argument can be built up employing only the three types of elementary statements as shown in Fig. 1.

We assume further that there are no circular arguments in it, consisting of a full directed circuit of edges. If one such occurs, as in Fig. 4(a), we avoid circularity by employing two terms such as $\underline{\underline{a}}_1$ and $\underline{\underline{a}}_2$ in Fig. 4(b), connected by the binary

vidya operator, to give $\underline{\underline{a}}_1 \vee \underline{\underline{a}}_2 = \underline{\underline{a}}$. We avoid also multiple

inputs into a term (say $\underline{\underline{a}}$) by using the vidya operator $\underline{\underline{a}}_1 \vee \underline{\underline{a}}_2 = \underline{\underline{a}}'$,

$\underline{\underline{a}}' \vee \underline{\underline{a}}_3 = \underline{\underline{a}}_2'$ etc. (See also Section 4 below). On the other

hand, there can be multiple outputs from a term to different connectives $\underline{\underline{Z}}_1$, $\underline{\underline{Z}}_2$ etc. Thus from the same term $\underline{\underline{a}}$ we can have

Eqs of the type $\underline{\underline{a}} \underline{\underline{I}} = \underline{\underline{p}}_1$, $\underline{\underline{a}} \underline{\underline{A}} \underline{\underline{h}} = \underline{\underline{p}}_2$, etc. On the other hand,

an operator, if it is unary can only have one input and one output,

and if it is binary can only have two inputs and one output.

For such an argument, the flow of logic in it can be expressed in the form of a set of equations, as shown, for instance, for Figs. 2, 3 and 4, and these equations are executable in sequence, using the given truth values of all inputs, so as to yield the truth values (in SIS) of all the outputs. The proof of

this is remarkably simple, as given below, using mathematical induction.

Firstly, it can be readily verified that all graphs of this nature, having upto 10 nodes are implementable. We use the term "implementable" to indicate that a set of SNS logical equations representing the graph can be written down, and executed in the sequential order in which they are listed. The input(s) for the algebraic equation representing any step contained in it would have been obtained in one of the preceding steps, or it (they) will be fresh inputs for the argument.

For larger graphs having n nodes, we prove the result by induction on the number n . We assume that all SNS logical graphs \mathcal{G}_m without circular arguments as subgraphs, with $m \leq n$ nodes, are implementable, and prove from this that all such graphs \mathcal{G}_{n+1} with $(n+1)$ nodes are implementable. The proof is as follows.

The graph \mathcal{G}_{n+1} has at least one output, say p . Then, this has arisen either from one term (say t_j) via a unary connection $\underline{\lambda}$ (as in Fig. 1(a)), or from two terms $\underline{t_k}$ and $\underline{t_l}$ via a binary connective $\underline{\lambda}$ or $\overleftarrow{\lambda}$ (as in Figs. 1(b) or 1(c)). In all cases, remove

the connective ($\underline{\underline{Z}}$ or $\overline{\underline{Z}}$) and the output term $\underline{\underline{p}}$, and obtain a graph \mathcal{G}_{n-1} , or \mathcal{G}_{n-2} , as the case may be. By assumption, this graph \mathcal{G}_{n-1} or \mathcal{G}_{n-2} is implementable.

Then, add the following equations at the end of the list of equations for the reduced graph (namely \mathcal{G}_{n-1} or \mathcal{G}_{n-2}):

$$\text{For } \mathcal{G}_{n-1} : \underline{\underline{tj}} \ \underline{\underline{Z}} = \underline{\underline{p}} \quad (22a)$$

$$\text{For } \mathcal{G}_{n-2} : \underline{\underline{tk}} \ \underline{\underline{Z}} \ \underline{\underline{tl}} = \underline{\underline{p}}, \text{ or } \underline{\underline{tk}} \ \overline{\underline{\underline{Z}}} \ \underline{\underline{tl}} = \underline{\underline{p}} \quad (22b,c)$$

It is quite obvious that the augmented list of equations for \mathcal{G}_{n+1} is implementable.

The above is an existential proof of the theorem that "Given the graph of any argument in PC, we can always write down for it a set of elementary SIS equations (of the types of (22a,b,c)), which can be executed in serial order". A Fortran program to perform the execution has been outlined, but not yet written. However, the above theorem indicates that the sequence of equations is theoretically capable of being written out. The discussion of the algorithm is reserved for a later publication, as it is common to SIS and QPL, and for arguments which use combinations of CL, SIS and QPL.

3. Logical graph of a QPL argument composed of Type-1 statements.

(a) Elementary statements of this type

A general statement of Type-1 (in QPL) has the form of Eq (1), and the three forms of elementary statements of this type are given in Eqs. (2), (3), (4) of Section 1. Since the variable x is the same throughout in an argument composed solely of Type-1 statements, we shall employ the following abbreviated notation for these:

$$\text{Unary} : \mathcal{V}(\underline{a} \underset{\zeta}{\sim} \underline{b}) \quad (23a)$$

$$\text{Binary forward} : \mathcal{V}(\underline{a} \underset{\zeta}{\sim} \underline{b} = \underline{c}) \quad (\text{Matrix}) \quad (23b)$$

$$\quad \quad \quad \underset{\equiv}{\underline{a}} \underset{\equiv}{\sim} \underset{\equiv}{\underline{b}} = \underline{c} \quad (\text{Boolean}) \quad (23c)$$

$$\text{Binary reverse} : \mathcal{V}(\underline{c} \underset{\zeta}{\sim} \underline{a} = \underline{b}) \quad (23d)$$

In these, $\mathcal{V} = V$ or $\}$ only; but ζ can be any one of the ten matrix operators in all three of (23a, b, d), while it can also be one of the two Boolean connectives, "upon" and "vidya", for the binary forward relation in (23c). When this occurs, the corresponding connectives IJ and IV of BA-3 become operative,

and the equations become

$$\underline{\underline{a}}_1 \vee \underline{\underline{a}}_2 = \underline{\underline{a}} ; \quad \underline{\underline{a}}_1 \wedge \underline{\underline{a}}_2 = \underline{\underline{a}} \quad (23e, f)$$

where $\underline{\underline{a}}_1$, $\underline{\underline{a}}_2$ stand for the canonical form in BA-3 of the two QPL inputs in the standard form. (See Part II, Section 6, for a description of these forms, and of the canonizer and standardizer which interconvert these representations of a QPL term. As an

illustration, $\forall(a_{1T}) \vee \exists(a_{2T}) \equiv (1 \ 0 \ 0) \otimes (1 \ 1 \ 0) = (1 \ 0 \ 0) \equiv \forall(a_T)$ and $\forall(a_{1T}) \vee \exists(a_{2F}) = (1 \ 0 \ 0) \otimes (0 \ 1 \ 1) = (0 \ 0 \ 0) = \emptyset(\underline{\underline{a}})$, the contradictory state for $\underline{\underline{a}}$.

It may be mentioned that the EPL contradictory state arises only as the output of the operator \vee , as is the case with \approx in SNS. It is also obvious that the variable x is the same for $\underline{\underline{a}}_1$, $\underline{\underline{a}}_2$ and $\underline{\underline{a}}$ in (23e,f). Upon inputting $\forall_{a_1}(\underline{\underline{a}}'_1) \equiv \underline{\underline{a}}'_1$ and $\forall_{a_2}(\underline{\underline{a}}'_2) \equiv \underline{\underline{a}}'_2$ in Eqs (23e) or (23f), the output $\underline{\underline{a}}'$ is obtained via the Boolean sum or Boolean product of the two 3-vectors, as the case may be.

We shall indicate how the quantifier \forall , and the connective

operator \underline{Z} , are treated in parallel channels, with a small

interaction of Z with \underline{Q}_y , when the QPL input(s) are put in (23a,b,c)

and the QPL output is worked out. The relevant formulae are given

in outline, along with the graphical flow chart representation

of the three forms of elementary statements, in Fig. 6.

Fig. 6. Elementary connective relations in QPL Type-1 sentences.

We shall illustrate the derivation of the formulae by taking the example of a categorical statement, discussed in detail in Part III [Ref. 1], which has the form $\underline{Q}_y(\underline{a} \ I = \underline{b})$. Suppose the input is $\cancel{\underline{Q}}_a(\underline{a}')$. Then, the flow of the logic on implementing Eq. (23a) with $\underline{Z} = \underline{I}$ to yield the output $\cancel{\underline{Q}}_b(\underline{b}')$ is best understood via the formulae in (24) below.

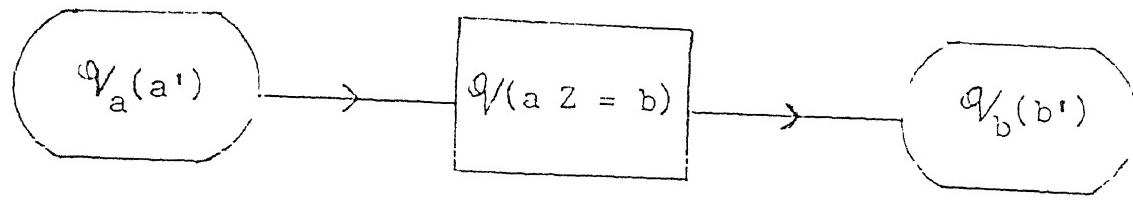
$$\cancel{\underline{Q}}_a(\underline{a}') , \cancel{\underline{Q}}_y(\underline{a} \ I) \mapsto \cancel{\underline{Q}}'_a(\underline{a}' \ I) \mapsto \cancel{\underline{Q}}_b(\underline{b}') \quad (24)$$

where

$$\cancel{\underline{Q}} \otimes \cancel{\underline{Q}}_a = \cancel{\underline{Q}}'_a , \cancel{\underline{Q}}'_a \oplus \cancel{\underline{Q}}_I = \cancel{\underline{Q}}_b \quad (25a,b)$$

and

$$\underline{a}' \ I = \underline{b}' , \cancel{\underline{Q}}_I = \exists \quad (25c,d)$$



$$\underline{a}' \approx \underline{b}' , (QV \oplus QV_a) \oplus QV_z = QV_b$$

$$QV_z = \forall \quad \text{for} \quad z = \text{E-type}$$

$$QV_z = \exists \quad \text{for} \quad z = \text{I-type}$$

$$\exists(\underline{a} A = \underline{b}) \equiv \exists(\underline{b} A = \underline{a}) \equiv \exists(\underline{a} I = \underline{b}) \equiv \exists(\underline{b} I = \underline{a})$$

For $\forall(a A = b)$, see Fig. 6(a)

Fig. 6(a) Unary QPL relation of Type-1

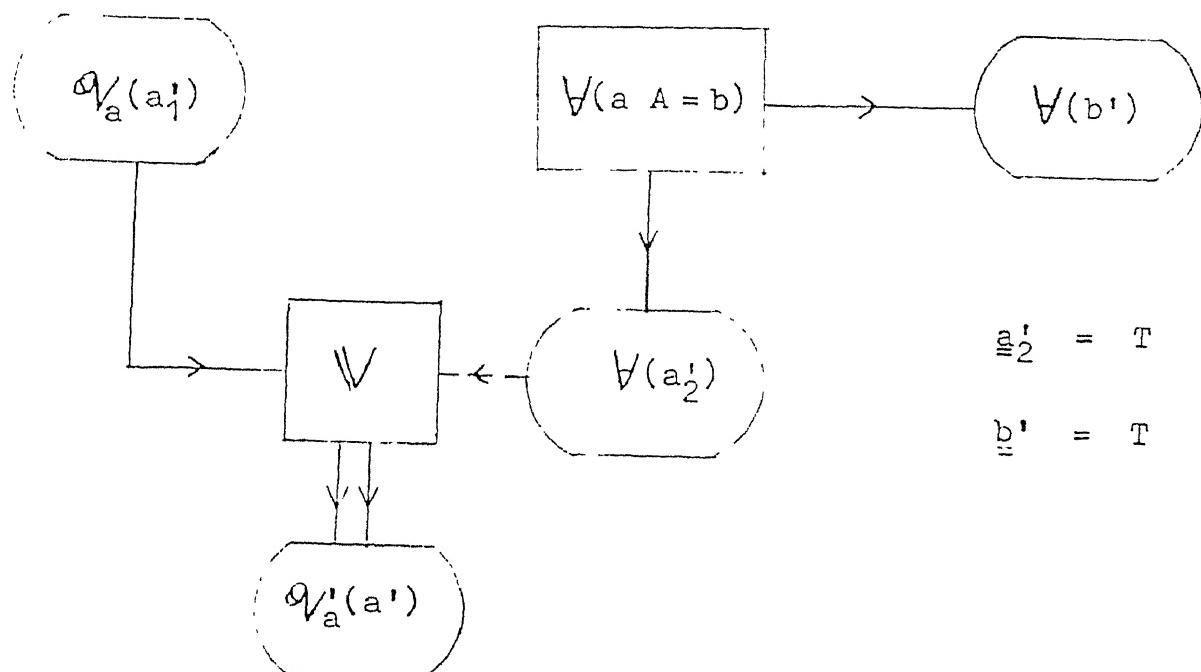
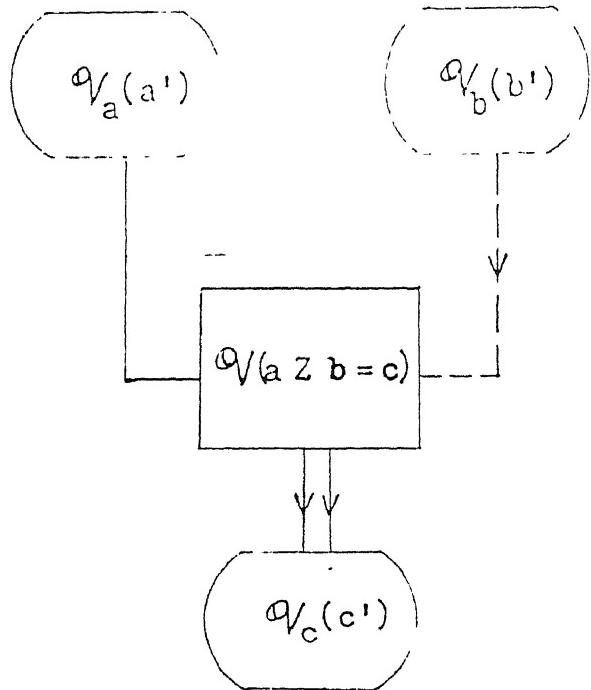


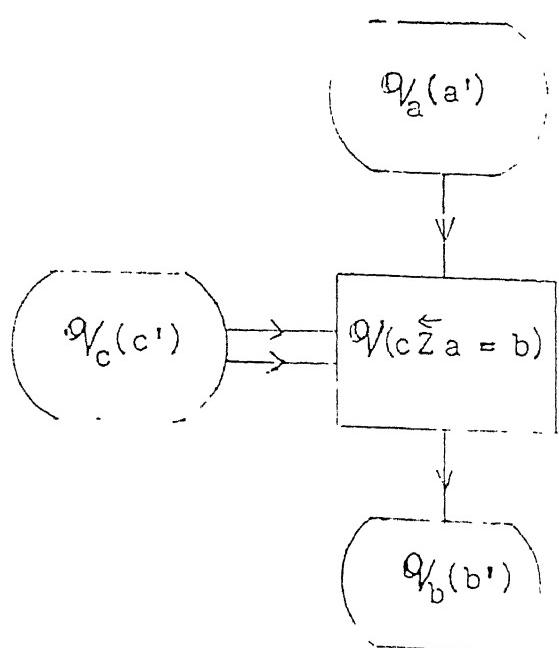
Fig. 6(b) Unary Type-1 relation with (\forall , A) and input $QV_a(\underline{a}')$. The modified A -construction is shown, with outputs $\forall(b')$ and $QV'_a(\underline{a}')$.



$$a' Z b' = c'$$

See text for formulae regarding \mathcal{V}_c for the three types of Z , namely A-type, O-type and E-type (Eqns. 30, 32, 33, 34).

Fig. 6(c) Binary forward QPL relation of Type-1.



$$a' Z = b', \text{ if } c' = T$$

$$a' Z^c = b', \text{ if } c' = F$$

$$(V \oplus V_c \oplus V_a) \\ \oplus (V_Z \text{ or } V_{Z^c}) = V_b$$

where \mathcal{V}_Z (or \mathcal{V}_{Z^c}) is the same as in Fig. 6(a), including the modified A-construction.

Fig. 6(d) Binary reverse QPL relation of Type-1.

An intuitive justification of the four equations (25a,b,c,d)

is as follows. The SNS equation (25c) is the simplest to see,

and it gives the output \underline{b}' for \underline{b} , given the input \underline{a}' for \underline{a}

and the operator equation $\underline{a} \otimes \underline{I} = \underline{b}$. As regards (25a), this employs

a new connective, which has been given the name "into" and having

the symbol \otimes , with all the quantifiers ($\forall, \forall_a, \forall'_a, \dots$)

having only the two states $\forall \equiv (1 \ 0 \ 0)$ and $\exists \equiv (1 \ 1 \ 0)$.

The property of this operator \otimes is best described by Table 3

which gives the multiplication table for the equation $\forall'_a = \forall \otimes \forall$

Table 3. Table for the "into" product $\forall \otimes \forall_a$

(b) Explanation of the new connective operator "into"

// The contents in Table 3 are simply explained as follows.

If the equation contains $\forall = \forall$, and the input quantifier \forall_a

is also \forall , then the effective quantifier \forall'_a in Eq (24) is seen

to be also necessarily \forall . If, on the other hand, either one of

\forall or \forall_a is \forall and the other is \exists , then only the one that is

Table 3. Table for the "into" product⁺ $\mathcal{V} \oplus \mathcal{V}_a$

\mathcal{V}	\mathcal{V}_a	A	E	D
\mathcal{V}	A	E	A	D
E		E	D	D
D		D	D	D

⁺The table lists the EPL states of $\mathcal{V}' = \mathcal{V} \oplus \mathcal{V}_a$, for the permissible input states of \mathcal{V} and \mathcal{V}_a in Sentences of Type-1.

logically less restrictive of the two (namely \exists) will be the quantifier of the resultant. This is because, if we take that the equation $\mathcal{V}(\underline{a} \underline{x} = \underline{b})$ is valid for $\mathcal{V} = \forall$, while the input state \mathcal{V}_a is only \exists , then the resultant equation giving the output $\mathcal{V}'_a (\underline{a}' \underline{x})$ will only be valid for $\mathcal{V}'_a = \exists$. In the same way, if the equation is valid for $\mathcal{V} = \exists$, while the input is with $\mathcal{V}_a = \forall$, then also the resultant \mathcal{V}'_a will only be \exists .

The fourth possibility, in which the equation has $\mathcal{V}_a = \exists$, and the input also has $\mathcal{V}_a = \exists$, requires special comment. The situation is best understood from Fig. 2 of Ref [1], in which the two inputs of \exists may, or may not, have any common elements. When they do not have any elements in common, the input $\mathcal{V}_a = \exists$, when fed into $\mathcal{V} = \exists$ for the equation, can lead to the resultant $\bar{\emptyset}$ ("for none") for $\mathcal{V}'_a = \mathcal{V} \oplus \mathcal{V}_a$. On the other hand, it is also possible for the resultant to be $\mathcal{V}'_a = \exists$, if the intersection is non-empty. Thus, the combination " \mathcal{V} into \mathcal{V}_a " in Eq. (25a)

can lead to either \exists or $\bar{\Phi}$. Consequently, the effective

quantifier for \mathcal{V}'_a is

$$\mathcal{V}'_a = \exists \oplus \bar{\Phi} = (1 \ 1 \ 0) \oplus (0 \ 0 \ 1) = (1 \ 1 \ 1) = \Delta \quad (26)$$

This is why the fourth entry, corresponding to $\mathcal{V} = \exists$, $\mathcal{V}_a = \exists$, in Table 3 is Δ .

The remaining entries in Table 3 are readily seen to arise from the following considerations. If one of either \mathcal{V} or \mathcal{V}_a is Δ , the resultant \mathcal{V}'_a in $\mathcal{V}'_{\underline{a} \underline{I}}$ is completely indefinite (Δ) and the fact that the other is \forall , \exists or Δ is irrelevant. Hence, we have the entries in the last row and column of Table 3.

It will be noticed that the entries in Table 3 are closely similar to those for the Boolean sum "upon" in $\mathcal{V} \oplus \mathcal{V}_a$. All of them (for $\mathcal{V} \oplus \mathcal{V}_a$) agree with the EPL state of $\mathcal{V} \oplus \mathcal{V}_a$, except one — namely $\exists \oplus \exists$, which is Δ , while $\exists \oplus \exists = \exists$. This new result, viz that $\exists \oplus \exists = \Delta$, derived here for $\underline{a} \underline{I} = \underline{b}$, is equally true for all matrix operators \underline{Z} in $\mathcal{V}_{\underline{a} \underline{Z} = \underline{b}}$, with the input $\mathcal{V}'_{\underline{a}'}$, leading to $\mathcal{V}'_{\underline{a}' \underline{Z}' = \underline{b}'}$, as will be shown

below. What is more, the operator \oplus is equally required for binary relations (both forward and reverse), as stated in the formulae listed in Fig. 6. It turns out that the connective \oplus plays a dominant role in the theory of not only QPL sentences of Type-1, but also in its extension to EPL sentences.

From the structure of Table 3, it is readily verified that \oplus is both associative and commutative. Further, it is also seen that the following distributive formulae between \oplus and \oplus are valid:

$$(V_1 \oplus V_2) \oplus V_3 = (V_1 \oplus V_3) \oplus (V_2 \oplus V_3) \quad (27a)$$

$$V_3 \oplus (V_1 \oplus V_2) = (V_3 \oplus V_1) \oplus (V_3 \oplus V_2) \quad (27b)$$

Boolean addition (\oplus) is, however, not distributive with respect to the "into" (\oplus) operation, so that, for example, sometimes

$$(V_1 \oplus V_2) \oplus V_3 \neq (V_1 \oplus V_3) \oplus (V_2 \oplus V_3) \quad (27c)$$

This can be verified from $V_1 = \forall$, $V_2 = \forall$, $V_3 = \exists$, when the l.h.s. gives \exists , while the r.h.s yields Δ .

The explanation of Eq. (25b) and the value \exists for \mathcal{V}_I in (25d) are both given below for the particular case of the connective operator $\underline{\underline{Z}} = \underline{\underline{I}}$. A general treatment for all connectives is given in the next section. It will be noticed that, in the expression $\mathcal{V}'_a (\underline{\underline{a}}' I)$, \mathcal{V}'_a can be one of \forall , \exists or Δ . We shall consider each of these in turn. As discussed in Part III, the equation $\forall(\underline{\underline{a}}' I = \underline{\underline{b}}')$ yields only the output $\exists(\underline{\underline{b}}')$ (and not $\forall(\underline{\underline{b}}')$), while the expression $\exists(\underline{\underline{a}}' I = \underline{\underline{b}}')$ also yields $\exists(\underline{\underline{b}}')$ as output (see Table 2, p 25 of Ref [1]). Thus, irrespective of \mathcal{V}'_a being \forall or \exists , \mathcal{V}_b is only \exists , for $\mathcal{V}'_a (\underline{\underline{a}}' I) = \mathcal{V}_b(\underline{\underline{b}}')$. On the other hand, if \mathcal{V}'_a is Δ , then it stands to reason that \mathcal{V}_b also will only have Δ . All these properties are readily satisfied by Eq (25b), with the condition $\mathcal{V}_I = \exists$, as given in Eq (25d). These two equations are empirical and have been designed to fit the properties of the logical formulation of a categorical implicational statement. However, it has been written in a form which is a special case of what is true for a unary QPL relation in general. This section only clarifies the method of approach and we shall give a full treatment for the general case in the next section.

(c) Justification of Fig.6 for all elementary connective relations in QPL.

(i) Binary relations for general $\tilde{\Sigma}$

We shall consider first Fig. 6(c) giving this type of elementary relation. Analogous to Eq. (24) the flow of logic may be pictured as follows. We are given the binary relation $\mathcal{V}(\underline{a} \tilde{\Sigma} \underline{b} = \underline{c})$ in which the inputs are $\mathcal{V}_a(\underline{a}')$ and $\mathcal{V}_b(\underline{b}')$, and we are required to find out both the quantifier and SNS states of $\mathcal{V}_c(\underline{c}')$.

$$\mathcal{V}_a(\underline{a}'), \quad \mathcal{V}_b(\underline{b}'), \quad \mathcal{V}(\underline{a} \tilde{\Sigma} \underline{b}) \mapsto \mathcal{V}_c(\underline{c}') \quad (28)$$

For this we first put in the two inputs "into" the corresponding places in the relation, and we obtain:

$$\mathcal{V}_a(\underline{a}') \otimes \mathcal{V} = \mathcal{V}'_a(\underline{a}') \quad (29a)$$

$$\mathcal{V}_b(\underline{b}') \otimes \mathcal{V} = \mathcal{V}'_b(\underline{b}') \quad (29b)$$

Thereafter the way in which these quantifiers go to form $\mathcal{V}_c(\underline{c}')$ is very complicated and requires separate treatment for the three types of matrix connectives, namely the $\tilde{\Sigma}$ -type, $\tilde{\Lambda}$ -type and $\tilde{\Omega}$ -type as in (30).

$$\tilde{\Sigma}\text{-type } (\tilde{\Sigma}, N = E^C) \quad (30a)$$

$$\tilde{\Lambda}\text{-type } (\tilde{\Lambda}, \tilde{\Omega}^C, \tilde{\Sigma}^C, \tilde{\Lambda}^C) \quad (30b)$$

$$\tilde{\Omega}\text{-type } (\tilde{\Omega}, \tilde{\Lambda}^C, \tilde{\Sigma}, \tilde{\Lambda}) \quad (30c)$$

After redefining \underline{a} and \underline{b} , the SNS relation can be converted to the form $\underline{a} \mathcal{Z} \underline{b} = \underline{c}$, with $\mathcal{Z} = E, A$, or O , for each of the three types of connectives in (30a, b, c). Then the formula for \underline{c}' is the same for all three, namely

$$\underline{a}' \mathcal{Z} \underline{b}' = \underline{c}' \quad (31)$$

The formula for \mathcal{V}_c on the other hand, is best given in the form of an algorithm for each case, as in (32), (33), (34) below.

$$\mathcal{Z} = A$$

$$\text{If } \underline{a}' = F, \underline{b}' = F, \text{ then } \underline{c}' = F \text{ and } \mathcal{V}_c = \mathcal{V}_a' \otimes \mathcal{V}_b' \quad (32a)$$

$$\text{If one of } \underline{a}', \underline{b}' \text{ is } F \text{ and the other is } T, \text{ then } \underline{c}' = F \text{ and } \mathcal{V}_c = \mathcal{V}_a' \text{ or } \mathcal{V}_b' \text{, according as } \underline{a}' \text{ or } \underline{b}' \text{ is } F \quad (32b)$$

$$\text{If both of } \underline{a}', \underline{b}' \text{ are } T, \text{ then } \mathcal{V}_c = \mathcal{V}_a' \oplus \mathcal{V}_b'. \text{ If however, both } \mathcal{V}_a \text{ and } \mathcal{V}_b \text{ are } \exists, \text{ then } \mathcal{V}_c = \mathcal{V}_a' \otimes \mathcal{V}_b' \quad (32c)$$

$$\mathcal{Z} = O$$

$$\text{If } \underline{a}' = T, \underline{b}' = T, \text{ then } \underline{c}' = T \text{ and } \mathcal{V}_c = \mathcal{V}_a' \otimes \mathcal{V}_b' \quad (33a)$$

$$\text{If one of } \underline{a}', \underline{b}' \text{ is } T \text{ and the other is } F, \text{ then } \underline{c}' = T \text{ and } \mathcal{V}_c = \mathcal{V}_a' \text{ or } \mathcal{V}_b' \text{, according as } \underline{a}' \text{ or } \underline{b}' \text{ is } T \quad (33b)$$

$$\text{If both } \underline{a}' \text{ and } \underline{b}' \text{ are } F, \text{ then } \mathcal{V}_c = \mathcal{V}_a' \oplus \mathcal{V}_b'.$$

$$\text{However, if } \mathcal{V}_a \text{ and } \mathcal{V}_b \text{ are both } \exists, \text{ then } \mathcal{V}_c = \mathcal{V}_a' \otimes \mathcal{V}_b'. \quad (33c)$$

E-type:

The formula for \underline{c}' is $\underline{a}' \sqcup \underline{b}' = \underline{c}'$ as in Eq (31) and the formula for the quantifier is as follows:

$$\mathcal{V}_a' \oplus \mathcal{V}_b' = \mathcal{V}_c' \quad (34a)$$

However, if both \underline{a}' and \underline{b}' have $\mathcal{V} = \exists$, then

$$\mathcal{V}_a' \oplus \mathcal{V}_b' = \mathcal{V}_c \quad (34b)$$

Note that there is a perfect symmetry between the formulae in Eq (32) for \sim and Eq (33) for \circlearrowleft and that what is valid for the input state T for \circlearrowleft is valid for F for \sim and vice versa. A phenomenon we had noticed in SNS theory of the relations involving the matrix operators for \circlearrowleft and \sim . We are not giving the proofs of the formulae in Eqs (32, 33 and 34) here, but shall give a short summary of the proof in Appendix 1.

Having thus considered all the ten SNS matrix connectives $\underline{\lambda}$ for Type-1 QPL binary forward statements, we shall now consider

.45a.

the two Boolean binary connectives \bigvee and \bigwedge . As mentioned

in Section 5(a), these connectives straightforwardly connect the

two QPL inputs $\mathcal{V}_{a1}(\underline{\underline{a}}^1)$ and $\mathcal{V}_{a2}(\underline{\underline{a}}^2)$ to give the output $\underline{\underline{a}}'$

in the form of a Boolean QPL relation in BA-3, namely

$\underline{\underline{a}}^1 \mathcal{Z} \underline{\underline{a}}^2 = \underline{\underline{a}}'$, where \mathcal{Z} is \bigvee or \bigwedge as the case may be. Hence

the procedure to be followed is to convert the standard form of

the inputs into the three-vector canonical form and then apply

the connective \bigvee (or \bigwedge) between them. These are implementable

using (35a,b), involving the Boolean sum and product, respectively,

of the two 3-vectors:

$$\bigvee: \underline{\underline{a}}^1 \oplus \underline{\underline{a}}^2 = \underline{\underline{a}}' ; \quad \bigwedge: \underline{\underline{a}}^1 \otimes \underline{\underline{a}}^2 = \underline{\underline{a}}' \quad (35a,b)$$

Of these, \bigwedge is extensively used also in connection with binary

reverse operators and with the unary forward connective relation

$\forall (\underline{\underline{a}} \approx \underline{\underline{b}}) -$ see Section 5(c) and 5(b) below.

(ii) Unary relations for all matrix connectives

We shall first consider all the ten matrix connectives occurring in a unary connective relation, although such a relation invariably arises from a binary reverse relation, as in Fig.6(d), having $\underline{c} = T$, or F , as the case may be. This is because we have exhaustively considered the unary connectives of the \underline{I} -type in Part III, and the considerations taken into account therein, and in Eqs (24) and (25) above, are all valid, and useful, for the discussion below in this section.

Firstly, we convert the four \underline{Q} -type connectives into the \underline{I} -type by the well-known relation

$$\neg \underline{a} \vee \underline{b} \equiv \underline{a} \Rightarrow \underline{b}, \text{ or } \underline{Q} \underline{N} = \underline{I} \quad (35)$$

This is because the relation actually takes effect as an implication, with all its properties as discussed in Part III.

However, the \underline{A} -relation has peculiar properties in its unary

form. With the existential quantifier, $\exists(\underline{a} \& \underline{b})$ has the significance of $\exists(\underline{a} \Rightarrow \underline{b})$ (as described in Part III), or

$$\exists(\underline{a} \underset{\sim}{A} = \underline{b}) \equiv \exists(\underline{a} \underset{\sim}{I} = \underline{b}) \quad (37a)$$

Since the matrix for $\underset{\sim}{A}$ is symmetric, $(\underline{a} \underset{\sim}{A} = \underline{b}) \equiv (\underline{b} \underset{\sim}{A} = \underline{a})$, which with the equivalence in (36a) gives

$$\exists(\underline{a} \underset{\sim}{A} = \underline{b}) \equiv \exists(\underline{b} \underset{\sim}{A} = \underline{a}) \equiv \exists(\underline{b} \underset{\sim}{I} = \underline{a}) \quad (37b)$$

Thus, we obtain the strange result (used extensively in Part III) that

$$\exists(\underline{a} \Rightarrow \underline{b}) \equiv \exists(\underline{b} \Rightarrow \underline{a}) \quad (37c)$$

although

$$\forall(\underline{a} \Rightarrow \underline{b}) \equiv \forall(\neg \underline{b} \Rightarrow \neg \underline{a}) \quad (37d)$$

In view of all these, we use Eqn (37a) to treat $\underset{\sim}{A}$ -connectives with the existential quantifier \exists as implications, and no further discussion of $\exists(\underline{a} \underset{\sim}{A} = \underline{b})$ is necessary. However, the elementary QPL statement $\forall(\underline{a} \underset{\sim}{A} = \underline{b})$, which arises from (or is equivalent to) the statements in (38) requires special discussion.

"We are given that $\forall(\underline{a} \& \underline{b})$ is true. Given $\mathcal{V}_{\underline{a}}(\underline{a}')$ what is $\mathcal{V}_{\underline{b}}(\underline{b}')?$ "

It will be noticed that the connective $\underline{\wedge}$ in $\underline{a} \underline{\wedge} \underline{b}$ asserts

that both \underline{a} and \underline{b} are true. Hence, if \underline{a}' is F, there can be a contradiction. Further, the state of \underline{b} (namely $\underline{b}' = T$)

is independent of what state \underline{a}' has. Extending these considerations to QPL, and including the \mathcal{Q}_V 's in (38), we obtain the flow chart of the logic for this particular case $\mathcal{V}(\underline{a} \underline{\wedge} \underline{b})$, as shown in Fig. 6(b).

The explanation of this figure is as follows. As mentioned above, the expression $\mathcal{V}(\underline{a} \underline{\wedge} \underline{b})$ asserts that both $\mathcal{V}(\underline{a})$ and $\mathcal{V}(\underline{b})$ are true; hence the output $\mathcal{V}(\underline{b}')$ with $\underline{b}' = T$. On the other hand $\mathcal{V}(\underline{a})$ may or may not be consistent with $\mathcal{Q}_a(\underline{a}')$. Hence, we designate these by $\mathcal{Q}_a(\underline{a}'_1)$ and $\mathcal{V}(\underline{a}'_2)$, and apply the vidya operator \mathcal{W} ^{to check} for consistency between them ^{and} to obtain the output $\mathcal{Q}_a'(\underline{a}')$. (As in the case of SMS logical graphs, we reserve the discussion of the consequences of the consistency test using \mathcal{W} for a later study. We only show that the statement

$\forall(\underline{a} \underline{A} = \underline{b})$ is implementable with whatever input that is provided.)

The discussion of the results for $\forall(\underline{a} \underline{E} = \underline{b})$, with

$\mathcal{V} = \forall \text{ or } \exists$, and input $\mathcal{V}_a(\underline{a}')$ as in Fig. 6(a) is simple. The net effective quantifier that goes in is obtained via the "into" operator as $\mathcal{V}'_a = \mathcal{V}_a \oplus \mathcal{V}$, exactly as for $\forall(\underline{a} \underline{I} = \underline{b})$.

However, the output, with \underline{E} , has the same quantifier as the input (being equivalent to it). Hence, for $\underline{Z} = \underline{E}$ in Fig. 6(a),

$$\underline{a}' \underline{E} = \underline{b}', \quad \mathcal{V}_a \oplus \mathcal{V} = \mathcal{V}_b \quad (39a, b)$$

However, for uniformity and convenience in programming, (39b) is written in the same form as for $\underline{Z} = \underline{I}$, namely

$$(\mathcal{V}_a \oplus \mathcal{V}) \oplus \mathcal{V}_E = \mathcal{V}_b, \quad \mathcal{V}_E = \forall \quad (39c)$$

We consider in the next subsection the algebraic treatment of the binary reverse relations which give rise to unary relations.

(iii) binary reverse relations for matrix connectives

This kind of relation, for statements of Type-1 in QPL, is described by the algebraic expression $\mathcal{V}(\underline{c} \underset{\leftarrow}{\gtrless} \underline{a} = \underline{b})$. The SNS expression within the bracket has the significance of Eq (13) in Part I (Ref. []), in that it asks for \underline{b}' , given $\underline{a} \underset{\leftarrow}{\gtrless} \underline{b} = \underline{c}$ and the inputs \underline{c}' and \underline{a}' . Now that the quantifier \mathcal{V} has been added, the inputs are $\mathcal{V}_c(\underline{c}')$ and $\mathcal{V}_a(\underline{a}')$, and we ask for the output $\mathcal{V}_b(\underline{b}')$, as shown in Fig. 6(d), for a binary ^{reverse}_A relation of Type-1.

In SNS, the information regarding the state of \underline{c}' leads to the equations

$$\underline{a}' \underset{\leftarrow}{\gtrless} \underline{b}' , \text{ if } \underline{c}' = T; \quad \underline{a}' \underset{\leftarrow}{\gtrless} \underline{b}'^C , \text{ if } \underline{c}' = F \quad (40)$$

which are seen to be readily capable of being taken over into QFL for Type-1 sentences (as written in Fig. 6(d)), since \mathcal{V} is common for \underline{a} , \underline{b} , \underline{c} and \underline{z} . If, however, the quantifier \mathcal{V}_c is different from \mathcal{V} , it is clear that $\mathcal{V}(\underline{c} \underset{\leftarrow}{\gtrless} \underline{a} = \underline{b})$ gets modified to

$$\mathcal{V}'(\underline{c}' \underset{\leftarrow}{\gtrless} \underline{a} = \underline{b}) , \text{ with } \mathcal{V}' = \mathcal{V} \oplus \mathcal{V}_c \quad (41a,b)$$

.51.

This gives rise to two QPL Type-1 equations analogous to (40), namely (42a) and (42b), with the inputs also mentioned in each case:

$$\mathcal{V}'(\underline{\underline{a}} \underset{\sim}{\underset{\sim}{z}} \underline{\underline{b}}) \text{ if } \underline{\underline{c}}' = T, \quad \mathcal{V}_a(\underline{\underline{a}}') \quad (42a)$$

and

$$\mathcal{V}'(\underline{\underline{a}} \underset{\sim}{\underset{\sim}{z}}^c \underline{\underline{b}}) \text{ if } \underline{\underline{c}}' = F, \quad \mathcal{V}_a(\underline{\underline{a}}') \quad (42b)$$

Equations (42a) and (42b) are identical in structure to the unary

QPL relation of Type-1 shown in Fig. 6(a). Hence, they lead to

the outputs \mathcal{V}_b and $\underline{\underline{b}}'$ as stated in Fig. 6(d). The formula for

$\underline{\underline{b}}'$ is clear, and the formula for \mathcal{V}_b arises from (41b) and the

formula for unary relations namely

$$(\mathcal{V}' \oplus \mathcal{V}_a) \oplus \mathcal{V}_z = \mathcal{V}_c \quad (43)$$

We may restate these here for ready reference. Given the inputs

$$\mathcal{V}_c(\underline{\underline{c}}'), \quad \mathcal{V}_a(\underline{\underline{a}}') \text{ in } \mathcal{V}(\underline{\underline{c}} \underset{\sim}{\underset{\sim}{z}} \underline{\underline{a}} = \underline{\underline{b}}) \quad (44a, b, c)$$

the output is

$$\mathcal{V}_b(\underline{\underline{b}}'), \text{ with } \underline{\underline{a}}' \underset{\sim}{\underset{\sim}{z}} \underline{\underline{b}}' \text{ for } \underline{\underline{c}}' = T \quad (44d)$$

$$\text{and } \underline{\underline{a}}' \underset{\sim}{\underset{\sim}{z}}^c \underline{\underline{b}}' \text{ for } \underline{\underline{c}}' = F \quad (44e)$$

and

$$(\mathcal{V}_c \oplus \mathcal{V}' \oplus \mathcal{V}_a) \oplus \mathcal{V}_z \text{ or } z^c = \mathcal{V}_b \quad (44f)$$

4. Structure and implementation of a general logical graph for Type-1 statements.

In this section, we shall consider the structure of a general graph representing the flow of logic in a QPL argument composed solely of Type-1 statements, and then show that such a graph is implementable with any set of QPL inputs, from the fact that each of the elementary statements of which it is composed is implementable.

We shall therefore comment on how to work out combinations of the elementary statements and then prove the general theorem that, for any finite logical graph, employing QPL statements of Type-1, the EFL states of the outputs can all be worked out in an algorithmic way — i.e. by following a sequence of operations in a stepwise fashion — which can therefore be computerized. Before proceeding to such combinations, we shall indicate the characteristics of the structure of a general QPL graph, following the lines adopted earlier for an SWS logical graph. It is to be noted that, although all inputs are QPL states, the intermediate terms and the outputs

may have any one of the EPL states and this particularly arises when they are outputs of the vidya operator.

(a) Preliminary steps for binary reverse connectives.

In a QPL graph, every unary operator has one input and one output in the relation $\mathcal{Q}(\underline{z} = \underline{b})$, which feature can also be generated by a binary reverse relation $\mathcal{Q}(\underline{c} \xrightarrow{\leftarrow} \underline{a} = \underline{b})$, with $\underline{c} = T$. In the latter case, there are two inputs, \underline{c}' and \underline{a}' , and the nature (truth value) of \underline{c}' generates the unary operator (\underline{z} or \underline{z}^c) connecting \underline{a} and \underline{b} in the unary chain. In view of the fact that the T and F states of \underline{c} , with $\underline{z} = A\text{-type}$ or $I\text{-type}$, can lead to either an $A\text{-type}$, or an $I\text{-type}$, relation, and since the further processing of the section of the graph dealing with this unary relation is quite different in the two cases, the first step in the unravelling of the logic of a QPL graph will be to work out the unary relations arising out of every binary reverse

relation. The steps involved in this are repeated in Eqs.(45a,b,c).

If $(\underline{c} \xrightarrow{\underline{Z}} \underline{a}) = \underline{b}$, and if $\underline{c}' = \mathcal{V}_c(T)$, then $\mathcal{V}'(\underline{a} \xrightarrow{\underline{Z}} \underline{b})$ (45a)

and if $\underline{c}' = \mathcal{V}_c(F)$, then $\mathcal{V}'(\underline{a} \xrightarrow{\underline{Z}} \underline{b})$ (45b)

where $\mathcal{V}' = \mathcal{V} \oplus \mathcal{V}_c$ (45c)

While doing this, if \underline{c}' is an independent original input, the rest of the manipulations mentioned in Eq (45) are carried out as such. If, however, it is not so, and it has an output of same step in the logical graph (argument) like $\mathcal{V}(\underline{g} \xrightarrow{\underline{Z}} \underline{c})$ leading to $\mathcal{V}_c(\underline{c}')$, then a " \underline{c} -construction" must be made as follows.

The term $\mathcal{V}_c(\underline{c}')$ is called $\mathcal{V}_{c1}(\underline{\underline{c}})$ and the input \underline{c}' in (45) is called $\mathcal{V}_{c2}(\underline{\underline{c}})$ and these two are connected by a vidya operator \mathbb{V} as shown in Fig. 7 to yield an output $\mathcal{V}_c(\underline{c})$.

Fig.7 Schematic diagram of \underline{c} -reconstruction

Thus effectively, the two equations

$$\mathcal{V}(\underline{g} \xrightarrow{\underline{Z}} \underline{c}) ; \quad \mathcal{V}(\underline{c} \xrightarrow{\underline{Z}} \underline{a} = \underline{b}) \quad (46a,b)$$

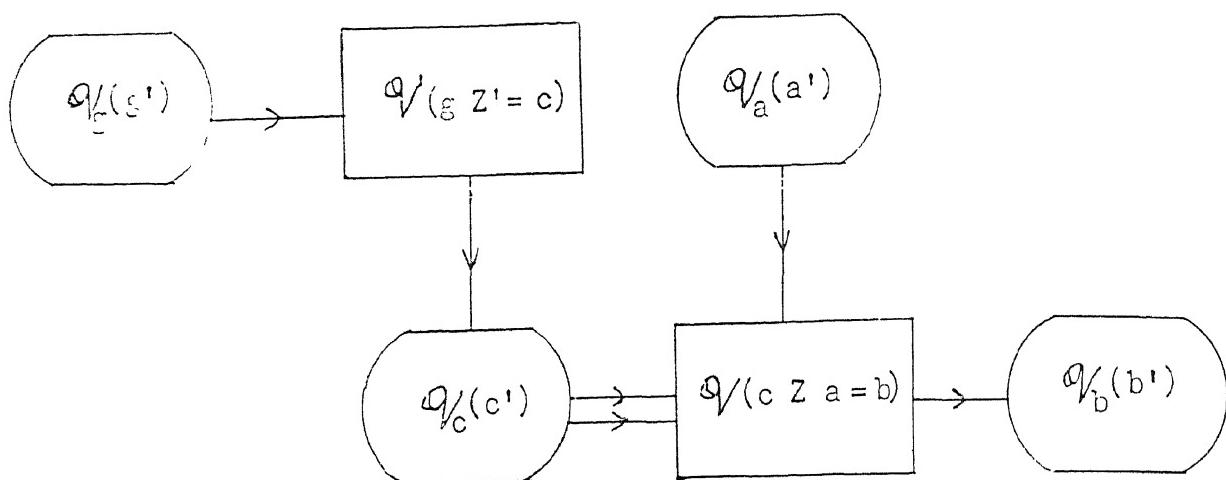


Fig. 7(a) Portion of graph before $\underline{\equiv}$ -reconstruction
representing Eqs. (46a,b)

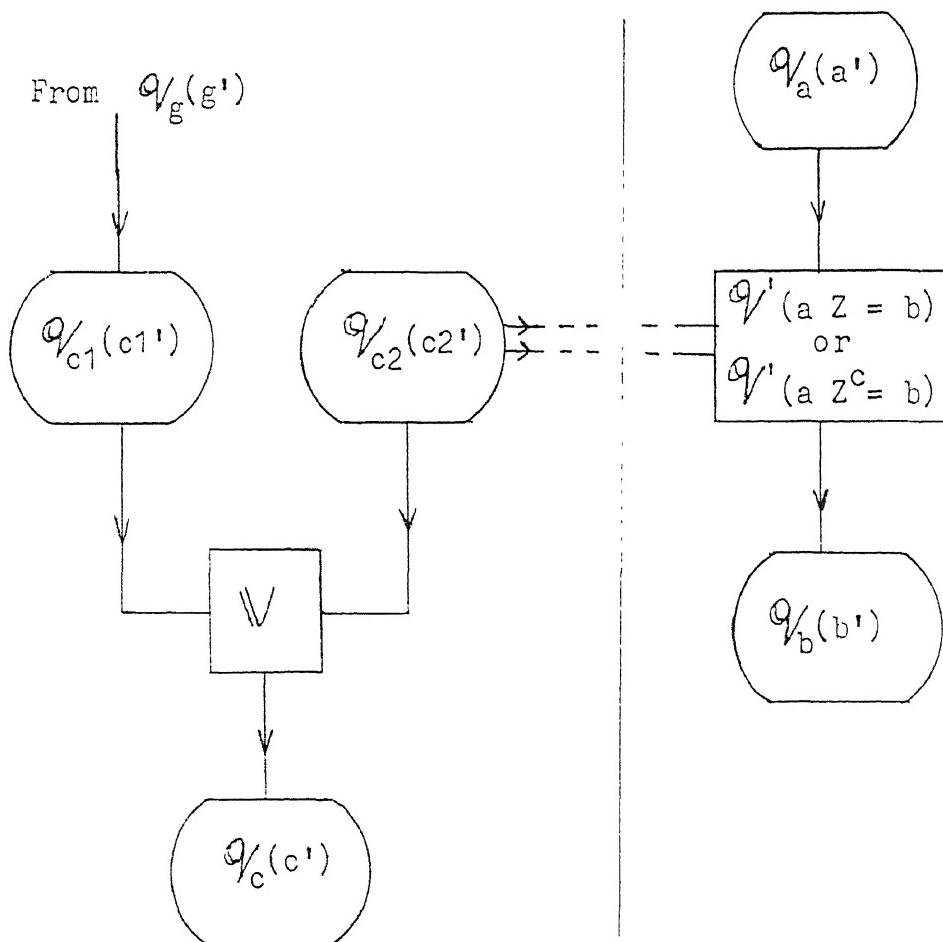


Fig. 7(b) Left: Part of the $\underline{\equiv}$ -reconstructed graph with the consistency check operator \mathbb{V} . Right: Unary relation between $\underline{\equiv}$ and $\underline{\equiv}$ generated by $Q'_c_2(c_2') \equiv Q'_c(c')$.

are converted into the set of equations

$$\mathcal{V}_g(\underline{\underline{c}} : \underline{\underline{z}} : = \underline{\underline{c}}_1') ; \quad \mathcal{V}(\underline{\underline{c}}_2 : \overset{\leftarrow}{\underline{\underline{z}}} \underline{\underline{a}}' = \underline{\underline{b}}') \quad (47a,b)$$

with the vidya relation

$$\underline{\underline{c}}_1' \vee \underline{\underline{c}}_2' = \underline{\underline{c}}' \quad (47c)$$

In this way all binary reverse relations are dispensed with and each one attached to a consistency check operator \vee and the graph is reduced to one having only unary forward and binary forward relations — the latter, however, containing both one of the ten matrix operators and one of the two Boolean operators, in particular, the vidya operator \vee .

(b) Preliminary steps related to A-type unary connectives

As mentioned above, the treatment of a unary forward relation when it has the general form $\forall(\underline{\underline{a}} : \underline{\underline{z}} = \underline{\underline{b}})$ requires the $\underline{\underline{A}}$ -construction as shown in Fig. 6b). Here, the unary $\underline{\underline{A}}$ -relation is taken back into the binary reverse $\underline{\underline{A}}$ -relation from which it arose and the logic of that is interpreted — namely that "For all $\underline{\underline{x}}$, both $\underline{\underline{a}}$ and $\underline{\underline{b}}$

are true". (As will be obvious, if the connective is any of the other three possible ones of the \mathcal{A} -type, a simple transformation of either \underline{a} , or \underline{b} , or both, by a negation brings it to the standard form $\forall(\underline{a} \mathcal{A} = \underline{b})$). Then, these two results \underline{a} is T and \underline{b} is T are applied independently. The first one, namely that "a is always true" is given by the expression $\forall(\underline{a}'_2)$, with $\underline{a}'_2 = T$, in Fig. 6(b). This is compared with the actual state of the QPL term $\mathcal{V}_a(\underline{a}'_1)$ by the consistency operator \mathbb{W} . On the other hand, the second result that " \underline{b} is true" is fed into the graph by means of the term $\forall(\underline{b}')$ with $\underline{b}' = T$, as shown in Fig. 6(b).

In this way, the graph in Fig. 6(c), for the case $\mathcal{V} = \forall$ and $\mathcal{Z} = \mathcal{A}$, has been modified so that no contradictions are fed from the \mathcal{A} -type relation into the graph. If any such arises, as can happen if $\underline{a}' = \underline{a}'_1 = F$, it will be in one of the final outputs, namely $\mathcal{V}'_a(\underline{a}')$, as shown in Fig. 6(b).

12) Consistency check for multiple inputs to a term

As mentioned in connection with SNS logical graphs, there can only be one edge in the graph, leading as input from a QPL term (say $\mathcal{V}_g(\underline{\underline{z}})$) to an operation (unary or binary) containing $\underline{\underline{z}}$ or $\overline{\underline{\underline{z}}}$; but more than one outputs, from two or more different operators, can lead to the same term. An example of three such is as follows:

$$\mathcal{V}_1(\underline{\underline{z}}_1 = \underline{\underline{p}}_1), \text{ yielding } \mathcal{V}_{\underline{\underline{p}}_1}(\underline{\underline{p}}_1') \quad (48a)$$

$$\mathcal{V}_2(\underline{\underline{z}}_2 \underline{\underline{h}}_2 = \underline{\underline{p}}_2), \text{ yielding } \mathcal{V}_{\underline{\underline{p}}_2}(\underline{\underline{p}}_2') \quad (48b)$$

$$\mathcal{V}_3(\underline{\underline{z}}_3 = \underline{\underline{p}}_3), \text{ yielding } \mathcal{V}_{\underline{\underline{p}}_3}(\underline{\underline{p}}_3') \quad (48c)$$

In (48a,b,c) $\underline{\underline{p}}_1$, $\underline{\underline{p}}_2$, $\underline{\underline{p}}_3$ all refer to the same QPL term $\underline{\underline{p}}$, but yield information coming from three different sources. Then,

it follows (as in SNS) that $\underline{\underline{p}}$ is the 3-vector that is obtained

by applying the vidya operator \mathbb{W} between $\underline{\underline{p}}_1$ and $\underline{\underline{p}}_2$ to obtain,

say, $\underline{\underline{p}}_2'$ and then between $\underline{\underline{p}}_2'$ and $\underline{\underline{p}}_3$ to obtain $\underline{\underline{p}}$ (see Eqns (49 a,b)).

$$\underline{\underline{p}}_1 \mathbb{W} \underline{\underline{p}}_2 = \underline{\underline{p}}_2', \quad \underline{\underline{p}}_2' \mathbb{W} \underline{\underline{p}}_3 = \underline{\underline{p}} \quad (49a,b)$$

The generalization to n outputs $\underline{\underline{p_1}}, \underline{\underline{p_2}}, \dots, \underline{\underline{p_n}}$ all leading to the same term $\underline{\underline{p}}$ is obvious. To reduce space, we shall write the final output after applying $(n-1)$ vidya operations to these in the form

$$\underline{\underline{p_1}} \vee \underline{\underline{p_2}} \vee \underline{\underline{p_3}} \dots \vee \underline{\underline{p_n}} = \underline{\underline{p}} \quad (50)$$

We shall call this process as the "V-construction".

We shall examine the EPL state of $\underline{\underline{p}}$ in general, first when

the inputs $\underline{\underline{p_1}}$ and $\underline{\underline{p_2}}$ belong to QPL or Δ (which occur in Part III —

and in the earlier sections of this paper so far), in $\underline{\underline{p_1}} \vee \underline{\underline{p_2}} = \underline{\underline{p}}$.
have

It must be remembered that, although we ~~need~~ only for the quantifiers

$\forall = \forall, \exists$ and Δ in $\mathcal{Q}(\underline{\underline{a}})$, the occurrence of $\underline{\underline{a}} = T$, or F , leads

to the following possibilities for $\underline{\underline{a}}$, namely $\forall \equiv (1 \ 0 \ 0)$,

$\exists \equiv (1 \ 1 \ 0)$, $\Delta = (0 \ 1 \ 1)$, $\bar{\Phi} = (0 \ 0 \ 1)$, $\Delta = (1 \ 1 \ 1)$.

A table of $\underline{\underline{a_1}} \vee \underline{\underline{a_2}} = \underline{\underline{a}}$ with these EPL states as inputs is shown in Table 4.

Table 4. List of the vidya product $\underline{\underline{a_1}} \otimes \underline{\underline{a_2}}$

Table 4. List of the vidya product $\underline{\underline{a}}_1 \otimes \underline{\underline{a}}_2$

$\underline{\underline{a}}_2$	A	E	Λ	\emptyset	Δ
$\underline{\underline{a}}_1$	(1 0 0)	(1 1 0)	(0 1 1)	(0 0 1)	(1 1 1)
A	A	A	\emptyset	\emptyset	A
E	A	E	Σ	\emptyset	E
Λ	\emptyset	Σ	Λ	\emptyset	Λ
\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset
Δ	A	E	Λ	\emptyset	Δ

$$\Sigma \equiv (010), \quad \emptyset \equiv (000)$$

It will be noticed that two new states (both belonging to EPL) occur for the vidya products $\underline{\underline{a}} \vee \underline{\underline{a}}$, of QPL terms $\underline{\underline{a}}$ and $\underline{\underline{a}}$, namely $\emptyset \equiv (0 \ 0 \ 0)$, standing for "contradictory" and $\zeta \equiv (0 \ 1 \ 0)$, for "some".

Of these, the former comes as a result of the consistency check made by \vee , and if $\underline{\underline{a}} \equiv \underline{\underline{a}} \vee \underline{\underline{a}} = (0 \ 0 \ 0)$, then it means that $\underline{\underline{a}}$ and $\underline{\underline{a}}$ are mutually inconsistent, as may be readily verified from the two pairs (for all, for none), and (for all, not for all), for which $\underline{\underline{a}} = \emptyset$. When this actually occurs in a graph, then any term to which it leads, via a path of unary or binary forward connectives, can be shown to have also the state \emptyset . This is readily verified for all ζ and $\overline{\zeta}$ in the elementary connective operations (except \vee).

The occurrence of ζ is even more restricted — it happens just for one combination, namely "There exists" vidya "Not for all"; but it must be remembered that this state is not necessarily

producible only by such a combination of QPL states. Instead,

as discussed in Part II, $\sum \Xi(0 \ 1 \ 0)$ is one of the three basic states of the EA-3 representation of the four QPL states \forall, \exists, Δ and Φ . That the state \sum can occur in very simple arguments is illustrated in the next subsection.

(d) Occurrence of the state \sum in a simple two-step argument containing the vidya operator.

We shall illustrate this by an actual pair of statements in mathematics. Suppose \mathcal{V} is any positive integer and \mathcal{T} a prime number. Then the following QPL statements are obviously true for the sentence $s(x) \equiv (\mathcal{V}(x) \Rightarrow \mathcal{T}(x))$, where x ranges over all positive integers.

$$(\exists x) (\mathcal{V}(x) \Rightarrow \mathcal{T}(x)) \equiv \exists(s_T) \equiv (1 \ 1 \ 0) \quad (51a)$$

$$(\exists x) (\mathcal{V}(x) \Rightarrow \neg \mathcal{T}(x)) \equiv \exists(s_F) \equiv \Delta(s_T) \equiv (0 \ 1 \ 1) \quad (51b)$$

Both (51a) and (51b) are QPL statements about the same sentence

s, and hence the common knowledge contained in both of them

is to be extracted by applying the vidya operator ∇ between them.

This gives

$$\begin{aligned} \exists(s_T) \nabla \exists(s_F) &\equiv (1 \ 1 \ 0) \otimes (0 \ 1 \ 1) = (0 \ 1 \ 0) \\ &\equiv \sum(s_T) \equiv \sum(s_F) \end{aligned} \quad (52)$$

The conclusion statement $\sum(s)$ thus obtained, stated in words, is that the

statement "If x is a positive integer, then it is a prime" is

true for some integers, but not for all integers (and obviously not for no integers).

The form of the above statement illustrating the symbolized logical term $\sum(s)$ is exactly the same as that asserted in the famous Gödel incompleteness theorem — namely that there are statements in predicate logic which are true, but which cannot be proved to be true for all x , or proved to be false for all x . As is well-known, Gödel has given

a carefully constructed theorem which has this property. However,

as will be seen from above, the Godel's state is identical with

the state $\sum = (0 \ 1 \ 0)$ in our extended predicate logic. As

mentioned above, the BA- $\bar{3}$ representation of quantified predicate

logic uses as basic states the following three $\bar{3}$ namely

$(1 \ 0 \ 0) \equiv$ For all, $(0 \ 1 \ 0) \equiv$ For some, $(0 \ 0 \ 1) \equiv$ For none.

The middle one "for some" excludes both "for all" and "for none";

and in this way three mutually exclusive states, or basic vectors,

of BA- $\bar{3}$ are constructed. Since EPL is based on BA- $\bar{3}$, there should

occur, under suitable circumstances, any number of examples ~~of~~ QPL

terms having the logical state $\sum(0 \ 1 \ 0)$, and the occurrence

of this does not indicate any "incompleteness" in the structure

of predicate logic. In fact, just as we found that the doubtful

state D(1 1) is a necessary consequence of the complete formulation

of propositional calculus with T(1 0) and F(0 1) alone, so

also both types of indeterminate states, namely $\Delta(1 \ 1 \ 1)$ and

$\Sigma(0 \ 1 \ 0)$ can occur, in addition to the well defined four states, when QPL is extended. When arguments are set up of various kinds, even if only the four QPL states are input, the intermediate terms and the output terms can have any one of the eight states of BA-3 of the extended formulation (EPL).

Thus, the unary and binary forward operators involving the matrix connectives (Σ) can lead only to five of the states, with Δ in addition to the four standard QPL states. We have now seen that two others — namely ϕ and Θ — arise from the application of the vidya operator. There only remains one out of the eight, namely $\Theta(1 \ 0 \ 1)$, standing for "For all or for none" but which can arise as $\forall \oplus \exists$, by the application of the operator "upon" (U). — which has not been introduced so far, \sqcap We shall consider this below, along with a discussion of how the four non-QPL states, $\Sigma, \Theta, \Delta, \phi$, can be incorporated in the general treatment of statements of Type-1. (For statements of Type-2 as discussed in Part II, all the eight states of BA-3 are completely taken into account in the matrix-algebraic treatment given in that Part. As will be discussed in the next section, statements of both Type-1 and Type-2 can occur together in an argument, and the algorithm for treating such arguments will be discussed in the next paper (Part V) of this series.)

(e) Incorporation of EPL states in Type-1 arguments

We have seen in the previous subsections of this Section 4 that the use of the four QPL canonical states \forall, \exists, Φ and Λ in arguments containing Type-1 statements could lead to one of the remaining states of EPL, namely $\Delta, \emptyset, \Sigma$. We shall therefore discuss the procedures for incorporating all the four additional states of EPL (including Θ) in such a Type-1 argument.

(i) Implementation of Δ as input

The state $\Delta \equiv (1 \ 1 \ 1) \equiv \exists \oplus \top$ is the completely indefinite state in QPL, which arises in the implementation of categorical statements (QPL implications), as shown in Part III. It is readily verified that, on putting this Δ as one of the inputs for any binary forward matrix connective operation, the output is also Δ , for all ten Z 's; so also for their unary relations. Also, it can be shown that, for $\underline{c} = \Delta$, corresponding to $\forall(\underline{c})$ in a binary reverse relation $\nabla(\underline{c} \xleftarrow{\underline{Z}} \underline{a} = \underline{b})$, the interpretation of D as $T \oplus F$ leads to an output Δ , as in Eqns (53a,b):

$$\underline{c}' = D, \quad \underline{c} \xrightarrow{\Sigma} \underline{a} = \underline{b} \rightarrow \underline{a}' \xrightarrow{\Sigma} \underline{a}' \oplus \underline{a}' \xrightarrow{\Sigma^C} \underline{b}' = \underline{a}' D = (1 \ 1) = D \quad (53a)$$

and

$$\mathcal{V}(\underline{b}') = \Delta \quad (53b)$$

Thus, Δ as an input in any step of a Type-1 QPL argument leads

to Δ as output for all matrix connectives. For the Boolean

connective $\text{I}\cup$ also $\Delta \oplus \underline{a} = \Delta$, so that Δ goes through as

to the output in the binary "upon" relation. Only for W is

$\Delta \otimes \underline{a} = \underline{a}$, for all states of \underline{a} , because $(1 \ 1 \ 1) \otimes$

$$(a_\gamma \ a_\delta \ a_\epsilon) = (a_\gamma \ a_\delta \ a_\epsilon).$$

Thus, the introduction of the non-QPL state Δ into Type-1 statements introduces no difficulty.

(ii) Implementation of Σ and Θ as inputs

In the last subsection (i), we interpreted $\mathcal{V}(D)$, where

\mathcal{V} is \vee or \exists , as equivalent to the EA-3 state $(1 \ 1 \ 1)$, corresponding

to Δ in LPL. In the case of Σ and Θ , $\Sigma(a_T) \equiv \Sigma(a_F)$ and

$\Theta(a_T) \equiv \Theta(a_F)$. This is because if only some (but not all)

have the property \underline{a} , some others will have the property $\neg \underline{a}$

so that we may reasonably associate Σ , when input into a Type-1 sentence, as giving a_D and hence having the properties of Δ .

(On the other hand, Σ as input for a Type-2 sentence can lead to quite definite and useful conclusions, as will be shown in Part V.)

In the same way, the sentence, "For all, or for none, \underline{a} is T", is the same as "For none, or for all, $\underline{a} = T$ ", which is equivalent to "For all, or for none, \underline{a} is F", so that $\Theta(\underline{a})$, when input into a Type-1 sentence, also has the properties of D for the SNS term \underline{a} , and hence, we may take it as having the properties of Δ when input into Type-1 sentences.

Hence, we assume that all types of uncertainties $\mathcal{Q}(a_D)$, in which \mathcal{Q} is any of the 7 quantifiers, other than \emptyset , of EFL, go in as $\mathcal{Q} \exists \Delta$ (indefinite) into the next step of the argument, if it is a Type-1 sentence, except for the operator \vee . As already mentioned, $\underline{a}_1 \vee \underline{a}_2 = \underline{a}$ is a sentence which can occur in both Type-1 and Type-2 arguments.

(iii) Implementation of \emptyset as input

Just as with Δ which has three 1's, ϕ has three 0's.

Hence, it also goes through as an impossible state \emptyset for the

output of all elementary connective operations for which it is

an input, except in $\underline{a_1} \cup \underline{a_2} = \underline{a}$. Hence, in a program ϕ as

input can be straightaway incorporated as an output state \emptyset .

(We do not explain the exception for \mathbb{W} , since we have not used

this logical connective "upon" anywhere. It can be shown that

it (i.e. the binary \sqcup relation) is relevant to the superposition

of information from "rumours"; but this discussion is deferred

to Part V.) Further, as stated before, ϕ can occur only as the

output of the binary \mathbb{W} -relation in our method of approach of

treats type-1 arguments.

We shall now proceed to the treatment of a chain of unary operations.

(f) Preliminary steps for unary chains

We consider a chain of unary relations connecting, say,
 \underline{a} with \underline{b} (or $\neg \underline{b}$), \underline{b} (or $\neg \underline{b}$) with \underline{c} (or $\neg \underline{c}$), ..., \underline{g} (or $\neg \underline{g}$)

with matrix connectives $\underline{z}_1, \underline{z}_2, \dots, \underline{z}_n$, and quantifiers

$\mathcal{V}_1, \mathcal{V}_2, \dots, \mathcal{V}_n$. We have already discussed the procedure

by which $\underline{\lambda}$ -type unary connectives can be made not $\underline{\lambda}$ form part of a

unary chain, so that we are left with only $\underline{\lambda}$ -type and $\underline{\exists}$ -type

connectives. Of these, $\underline{\lambda}$ -type relations have been exhaustively

considered in Part III, and it has been shown that the above

chain of n implications can be reduced to be equivalent to a single

Type-1 implicational statement connecting \underline{a} with \underline{g} . In saying

this, we consider the "invalid", or Δ -state, conclusion also to be

a single implication, for this purpose, and write it as $\mathcal{V}(\underline{a} \underset{\underline{\lambda}}{\sim} \underline{b})$.

For working out n implications like the above in sequence,

we cannot input $\mathcal{V}'(\underline{a}')$ and work out $\mathcal{V}'(\underline{b}')$ etc sequentially

in an algorithmic manner, one after the other in succession.

Instead, the n Type-1 implications have to be "theoretically"

combined to get \mathcal{V} and \underline{z} in $\mathcal{V}(\underline{a} \underline{z} = \underline{g})$, from $\mathcal{V}_1, \underline{z}_1;$

$\mathcal{V}_2 \underline{z}_2; \dots, \mathcal{V}_n, \underline{z}_n$; and thereafter \mathcal{V}_a and \underline{a}' can be input to

$\mathcal{V}(\underline{a} \underline{z} = \underline{g})$ for obtaining the output \mathcal{V}_g and \underline{g}' of the unary

chain. Because of this, we shall give below, in outline, the

formulae for combining two unary Type-1 implications to obtain

the resultant of the same type, and then show how the formulae

can be iterated.

(i) Combination of two or more Type-1 implications

We first consider two QPL implications of Type-1 in succession namely $\mathcal{V}_1(\underline{a} \Rightarrow \underline{b})$ or $\mathcal{V}_1(\underline{a} \Rightarrow \neg \underline{b})$ and $\mathcal{V}_2(\underline{b} \Rightarrow \underline{c})$ in succession, where $\mathcal{V} = \forall$ or \exists . The resultant $\mathcal{V}_3(\underline{a}(\text{or } \neg \underline{a}) \Rightarrow \underline{b}(\text{or } \neg \underline{b}))$ for all these cases is summarized in Table 5. By putting $\underline{a} = \neg \underline{a}$ and $\underline{c} = \neg \underline{c}$, all possible combinations which occur in practice can be obtained from the above eight possibilities.

Table 5. Resultant of two Type-1 implications in succession.

Table 5. Resultant of two Type-1 implications in succession

Sl. No.	First implication	Second implication	Resultant	Name of categoric syllogism
1	$\forall(\underline{\underline{a}} \underline{\underline{I}} = \underline{\underline{b}})$	$\forall(\underline{\underline{b}} \underline{\underline{I}} = \underline{\underline{c}})$	$\forall(\underline{\underline{a}} \underline{\underline{I}} = \underline{\underline{c}})$	<u>Barbara</u>
2	$\forall(\underline{\underline{a}} \underline{\underline{I}} = \neg \underline{\underline{b}})$	$\forall(\underline{\underline{b}} \underline{\underline{I}} = \underline{\underline{c}})$	$\forall(\underline{\underline{a}} \underline{\underline{D}} = \underline{\underline{c}}) \equiv \Delta$	Invalid
3	$\forall(\underline{\underline{a}} \underline{\underline{I}} = \underline{\underline{b}})$	$\exists(\underline{\underline{b}} \underline{\underline{I}} = \underline{\underline{c}})$	$\Delta(\underline{\underline{a}} \underline{\underline{I}} = \underline{\underline{c}}) \equiv \Delta$	Invalid
4	$\forall(\underline{\underline{a}} \underline{\underline{I}} = \neg \underline{\underline{b}})$	$\exists(\underline{\underline{b}} \underline{\underline{I}} = \underline{\underline{c}})$	$\exists(\neg \underline{\underline{a}} \underline{\underline{I}} = \underline{\underline{c}})$	<u>Dimension</u>
5	$\exists(\underline{\underline{a}} \underline{\underline{I}} = \underline{\underline{b}})$	$\forall(\underline{\underline{b}} \underline{\underline{I}} = \underline{\underline{c}})$	$\exists(\underline{\underline{a}} \underline{\underline{I}} = \underline{\underline{c}})$	<u>Darii</u>
6	$\exists(\underline{\underline{a}} \underline{\underline{I}} = \neg \underline{\underline{b}})$	$\forall(\underline{\underline{b}} \underline{\underline{I}} = \underline{\underline{c}})$	$\exists(\underline{\underline{a}} \underline{\underline{D}} = \underline{\underline{c}}) \equiv \Delta$	Invalid
7	$\exists(\underline{\underline{a}} \underline{\underline{I}} = \underline{\underline{b}})$	$\exists(\underline{\underline{b}} \underline{\underline{I}} = \underline{\underline{c}})$	$\Delta(\underline{\underline{a}} \underline{\underline{I}} = \underline{\underline{c}}) \equiv \Delta$	Invalid
8	$\exists(\underline{\underline{a}} \underline{\underline{I}} = \neg \underline{\underline{b}})$	$\exists(\underline{\underline{b}} \underline{\underline{I}} = \underline{\underline{c}})$	$\Delta(\underline{\underline{a}} \underline{\underline{D}} = \underline{\underline{c}}) \equiv \Delta$	Invalid

⁺In Table 4 of Part III. Note the occurrence of the "New Type" of syllogism, Dimension (IEI'-1)

^{*}For all "invalid" resultants, the output $\underline{\underline{c}}$ is Δ , irrespective of the QPL input state $\underline{\underline{a}}$,

It will be noticed that only three combinations in Table 5 lead to valid conclusions for the resultant. Of these, Barbara and Darii are classical, but we have, in addition, one of the new types of syllogisms derived in Part III, namely Dimension (IEI'-1). This can only be worked out by reversing the séquence of the first two implications and then reversing, once again, the valid resultant thus obtained. However, in the form in which it is stated in Table 5, it is algorithmically applicable for working out the valid resultants of sequences of n implications, and this leads to the same results as those shown in Section 5 of Part III. Thus, the forward sequence $\forall(\neg \underline{a} \underline{\mathcal{I}} \Rightarrow \neg \underline{b})$, $\exists(\underline{b} \underline{\mathcal{I}} \Rightarrow \underline{c})$, $\forall(\underline{c} \Rightarrow \underline{d})$ can be worked out as follows to yield $\exists(\underline{a} \Rightarrow \underline{d})$:

$$\forall(\neg \underline{a} \underline{\mathcal{I}} \Rightarrow \neg \underline{b}), \quad \exists(\underline{b} \underline{\mathcal{I}} \Rightarrow \underline{c}) = \exists(\underline{a} \Rightarrow \underline{c}) \quad (\text{Analogue of } \underline{\text{Dimension}}) \quad (54a)$$

$$\exists(\underline{a} \Rightarrow \underline{c}), \quad \forall(\underline{c} \Rightarrow \underline{d}) = \exists(\underline{a} \Rightarrow \underline{d}) \quad (\text{Analogue of } \underline{\text{Darii}}) \quad (54b)$$

(ii) Inclusion of E-type unary connectives

Since A-type Type-1 statements, with both \forall and \exists as quantifier, do not occur in unary chains, we need consider only the E-type connective relations that may occur in between implications in a unary chain. Considering first combinations of two E-type relations, the following equations are obvious:

$$\mathcal{V}_1(\underline{\underline{a}} \underline{\underline{z}}_1 = \underline{\underline{b}}), \quad \mathcal{V}_2(\underline{\underline{b}} \underline{\underline{z}}_2 = \underline{\underline{c}}) = \mathcal{V}_3(\underline{\underline{a}} \underline{\underline{z}}_3 = \underline{\underline{c}}) \quad (55a)$$

where

$$\mathcal{V}_1 \oplus \mathcal{V}_2 = \mathcal{V}_3; \quad z_1 z_2 = z_3 \quad (55b)$$

with

$$\mathcal{V}_1, \mathcal{V}_2 = \forall, \exists \quad \text{and} \quad z_1, z_2 = \underline{\underline{E}} \text{ or } \underline{\underline{E}}^C (\equiv \underline{\underline{N}}) \quad (55c)$$

Combinations of an I-type and an E-type relations also follow simple rules, as in (56) and (57) below.

$$\mathcal{V}_1(\underline{\underline{a}} \underline{\underline{z}} = \underline{\underline{b}}), \quad \mathcal{V}_2(\underline{\underline{b}} \underline{\underline{z}}_2 = \underline{\underline{c}}) = \mathcal{V}_3(\underline{\underline{a}} \underline{\underline{z}} = \underline{\underline{x}}) \quad (56a)$$

where

$$(\mathcal{V}_1 \oplus \exists) \otimes \mathcal{V}_2 = \mathcal{V}_3 \quad (56b)$$

and

$$\underline{\underline{x}} = \underline{\underline{c}} \quad \text{if} \quad z_2 = \underline{\underline{E}}, \quad \text{and} \quad x = \neg \underline{\underline{c}} \quad \text{if} \quad z_2 = \underline{\underline{E}}^C \quad (56c)$$

$$\mathcal{V}_1(\underline{\underline{a}} \underline{\underline{z}}_1 = \underline{\underline{c}}), \quad \mathcal{V}_2(\underline{\underline{b}} \underline{\underline{z}}_2 = \underline{\underline{c}}) = \mathcal{V}_3(\underline{\underline{y}} \underline{\underline{z}}_3 = \underline{\underline{c}}) \quad (57a)$$

where

$$(\mathcal{V}_1 \oplus \mathcal{V}_2) \oplus \mathcal{V}_3 = \mathcal{V}_3 \quad (57b)$$

and

$$y = \underline{\underline{a}}, \text{ if } \underline{\underline{z}}_1 = E, \text{ and } y = \neg \underline{\underline{a}}, \text{ if } \underline{\underline{z}}_1 = \mathcal{N}(E^C) \quad (57c)$$

To work out a chain of $\underline{\underline{I}}$ -type or $\underline{\underline{E}}$ -type relations, we consider their associativity.

(iii) Associativity of unary connective relations in a unary path.

This can be proved quite generally. For this, we first take three unary relations $\mathcal{V}_1(\underline{\underline{a}} \underline{\underline{z}}_1 = \underline{\underline{b}})$, $\mathcal{V}_2(\underline{\underline{b}} \underline{\underline{z}}_2 = \underline{\underline{c}})$ and $\mathcal{V}_3(\underline{\underline{c}} \underline{\underline{z}}_3 = \underline{\underline{d}})$ and denote the equivalent single unary relation by $\mathcal{V}(\underline{\underline{a}} \underline{\underline{z}} = \underline{\underline{d}})$.

For convenience, we write it as a product of three operators

$$(\mathcal{V}_1, z_1) = z_1 \text{ (say)}, \quad (\mathcal{V}_2, z_2) = z_2 \text{ (say)} \text{ and } (\mathcal{V}_3, z_3) = z_3 \text{ (say)}$$

in Eq (58a and b), with brackets in the two different ways of association:

$$((\mathcal{V}_1, z_1) (\mathcal{V}_2, z_2)) (\mathcal{V}_3, z_3) \equiv (z_1 z_2) z_3 = z \text{ (say)} \quad (58a)$$

$$(\mathcal{V}_1, z_1) ((\mathcal{V}_2, z_2) (\mathcal{V}_3, z_3)) \equiv z_1 (z_2 z_3) = z' \text{ (say)} \quad (58b)$$

By taking the data in Table 7, and in Eqns (55), (56) and (57), it is readily verified that for all \mathcal{V} 's being \forall and \exists , and all Z 's being I , E or N , if

$$z \equiv (\mathcal{V}, Z) \text{ and } z' \equiv (\mathcal{V}', Z') ; \quad z \equiv z' \text{ in (58a,b)} \quad (59)$$

In other words, unary connective relations in succession are associative. (The detailed proof of ~~associativity~~ is long, but quite simple, and it is omitted).

(iv) Implementation of a unary bouquet

We have already seen that there cannot be more than one incoming edge to a term from an operator (for shortness, we use the symbol t for a term $\mathcal{V}(\underline{t})$ and Z for an operator (\mathcal{V}, Z) producing a unary or binary relation). However, more than one edge can connect a term t_1 to different operators (Z_1, Z_2, Z_3 , say). Hence, the unary paths emerging from a term t_1 can have a "bouquet" structure, as shown in Fig. 8(a).

Fig.8. Reduction of a general bouquet of unary paths to a bouquet of reduced single unary relations.

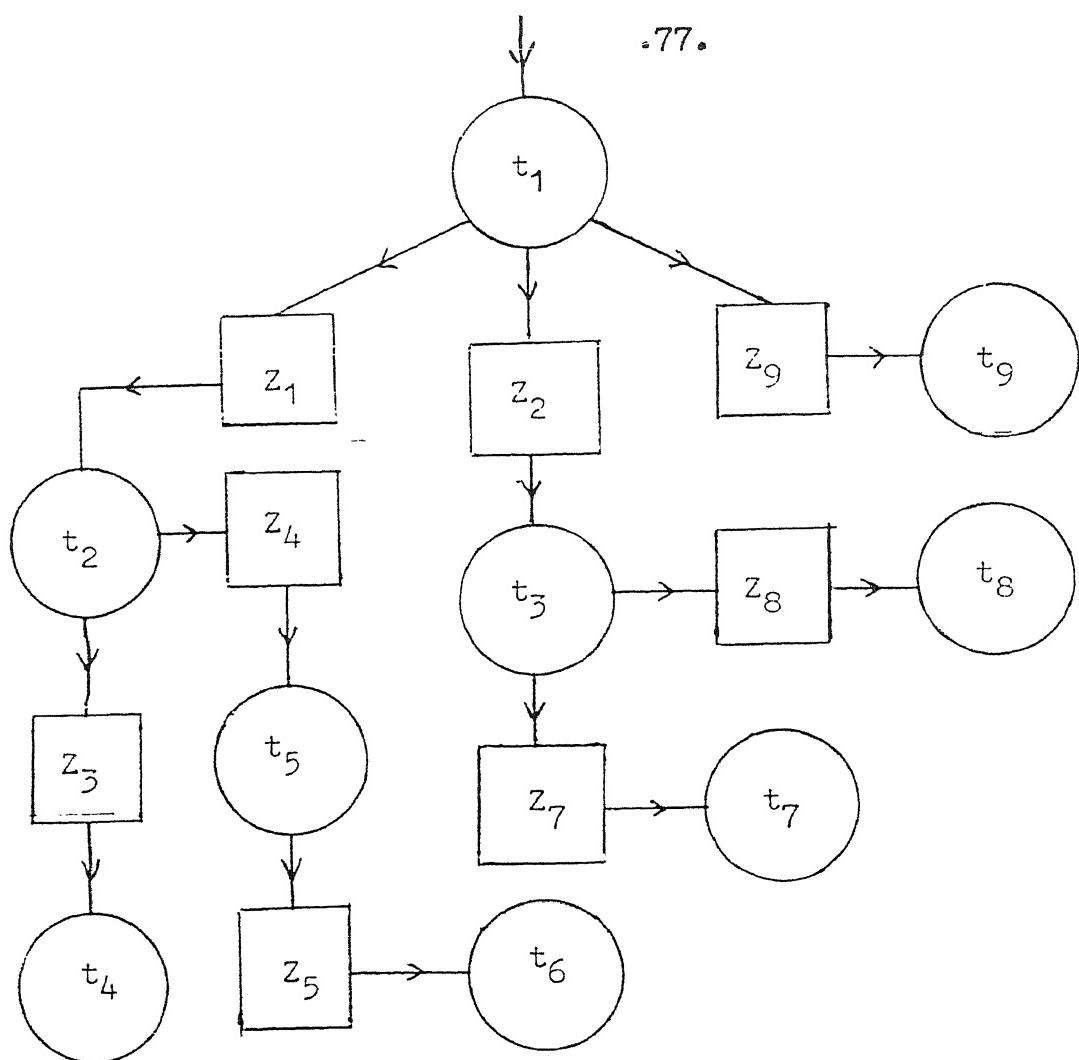


Fig. 8(a) Bouquet of edges starting from t_1 along unary paths and ending in t_4 , t_6 , t_7 , t_8 and t_9 .

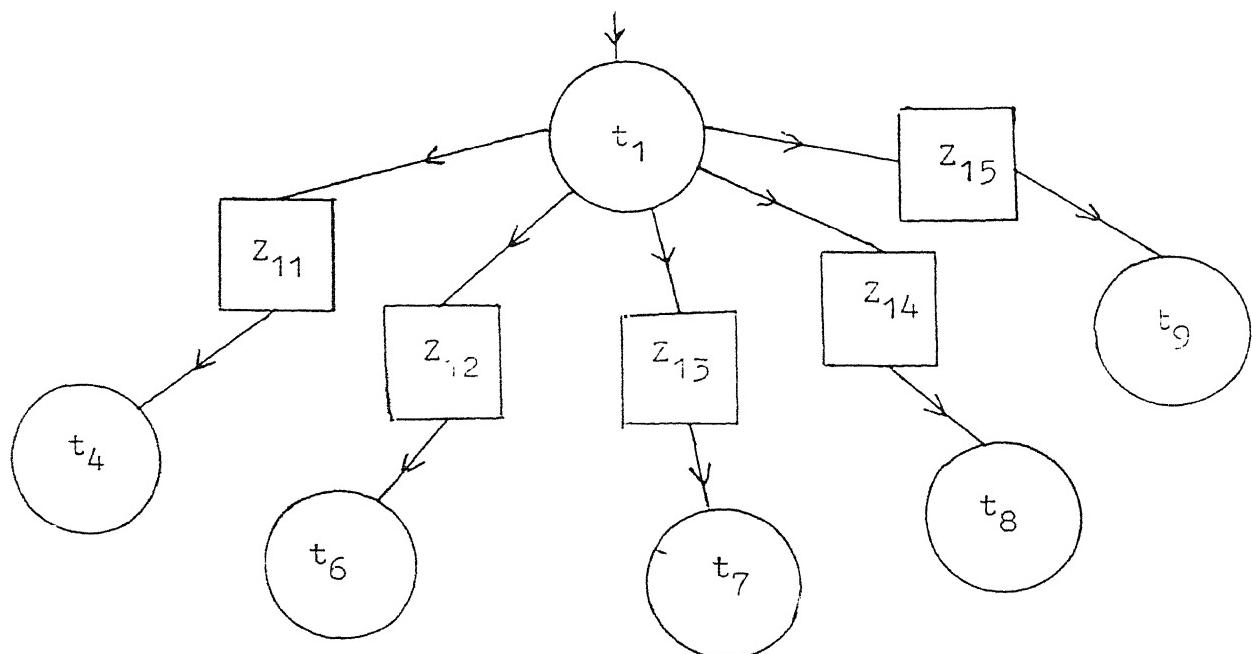


Fig. 8(b) Equivalent single connective unary paths, all starting from t_1 and ending in t_4 , t_6 , t_7 , t_8 and t_9 .

The word "bouquet" is suggestive of t_1 forming the stalk, and the different directed paths from it forming branching stems, leading to "flowers" (t_4, t_6, t_7, t_8, t_9) as termini of the various unary paths from t_1 . It is ^a simple theorem in the theory of graphs that form trees (of which bouquets are a special class) that two vertices are connected by a unique path in a tree. Hence, the term forming the stalk t_1 is connected by a unique unary path to each flower, forming its terminus.

We now show, using the theory developed earlier in this section 4(f), that a general bouquet of unary paths of Type-1 relations emerging from t_1 , as in Fig. (8a), is reducible to a set of single Type-1 unary relations from t_1 straight to the terminus of each of the unary paths. This follows from the fact that we can collapse n unary QPL relations (of Type-1, containing only \exists -type and \forall -type connectives) into a single relation of the same type. The proof follows from two successive relations being reducible

to a single one, which can be applied iteratively to any number.

The associativity of this process, stated above, leads to the uniqueness of the single QPL Type-1 relation.

Thus, using the contracted notation mentioned in Section 4(f) (iii), if we follow the unary path from t_1 to t_2 , t_5 and t_6 , then

$$t_1 Z_1 = t_2, \quad t_2 Z_4 = t_5, \quad t_3 Z_5 = t_6 \quad (60a)$$

so that

$$t_1 Z_{12} = t_6, \text{ where } Z_{12} = Z_1 Z_4 Z_5 \quad (60b)$$

The relation (60b) is shown in Fig.8(b). Similarly, each one of

the paths from t_1 to t_2 , t_7 , t_8 , t_9 can be reduced to single

unary relations by following the paths from its origin to its

terminus. Thus, Fig. 8(a) is reduced to Fig. 8(b), using the

unary operators Z_{11} , Z_{12} , Z_{13} , Z_{14} , Z_{15} , one at a time for

obtaining the output term at each terminus, with $Z_{11}=Z_1 Z_3$, $Z_{12}=Z_1 Z_4 Z_5$, $Z_{13}=Z_2 Z_7$, $Z_{14}=Z_2 Z_8$, $Z_{15}=Z_9$.

Thereafter, the input (say $V_1(t_1)$) can be fed to each relation

and the outputs $\mathcal{V}'_{\underline{t}_4}(t'_4)$, etc can be worked out. These latter will be either final outputs, or will be fed to binary relations, which are algorithmically implementable from input(s) to output. Thus, by the "bouquet reduction" procedure, the general graph of an argument is converted to a form in which it consists only of single unary relations, connecting input and/or output terms of binary relations. This reduced graph has only relations of the types in (61a), written in the abbreviated version

$$t_1 Z = t_2 \quad \text{or} \quad t_1 Z t_2 = t_3 \quad (61a)$$

Eqns (61) stand for the Type-1 relations

$$\mathcal{V}(\underline{t}_1 Z = t_3) \text{ and } \mathcal{V}(\underline{t}_1 Z \underline{t}_2 = \underline{t}_3) \quad (61b)$$

In the next section, we shall summarize the procedures to be adopted for obtaining such a reduced graph from a general set of Type-1 relations forming an argument and then prove that the reduced graph is always implementable for Type-1 statements, using an argument analogous to that adopted for SNS statements.

5. Summary of the implementation procedure for a Type-1 argument

We assume that the logical graph of the argument is available in terms of the set of unary, binary forward and binary reverse relational statements between the various terms t_j via operators Z_j . We shall first describe the various steps by which the argument can be worked out and the conclusions corresponding to the output terms can be determined when the input terms are given. The steps are also given briefly in the form of a flow chart in Fig. 9 A to I. The descriptions, in paras A to I below, follow the same order.

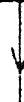
Fig.9. Outline of Flow Chart.

A. First we search for the input and output terms and classify all terms as original inputs ($t_j^{(i)}$), final outputs ($t_j^{(o)}$) and intermediate (middle) input-outputs ($t_j^{(m)}$).

Fig.9. Outline of Flow Chart

($t^{(i)}$ = input term, $t^{(m)}$ = middle term, $t^{(o)}$ = output term;
 $t^{(ai)}$, $t^{(am)}$, $t^{(ao)}$ = "additional" input, middle,
output terms; Z = Connective operator)

A. Write down the elementary steps in the argument
in any order. Check and list the terms as $t_j^{(i)}$,
 $t_j^{(m)}$ and $t_j^{(o)}$. Verify that the graph has no
directed circuits. If it has, then go to B.



B. Check for occurrence of directed circuits. For
each one present, make the "circuit removal
construction" as in Fig.4(b) for each circuit.
Add the vertices corresponding to new terms $t^{(i)}$,
 $t^{(m)}$, $t^{(o)}$ and the new operators Z_j (of type ∇)
to the list in A.



C. Make the ∇ -construction for multiple inputs to
a term t_j , as in Eqs.(49) and (50), and do this
for all such terms. Add the additional Z_j (of
type ∇) and the $t_j^{(m)}$ and $t_j^{(o)}$ so generated to
the list in A.



Fig.9. Outline of Flow Chart (Contd.)

D. List the binary reverse relations. If one of them, \underline{c} is not a $t^{(i)}$, carry out a \underline{c} -reconstruction as in LIS of Fig.7(b). Add or replace the operators Z_j so produced and add the additional terms $t^{(ai)}$, $t^{(ao)}$ and also replace the relevant $t^{(m)}$ in the list in A.

E. Input the QPL states of $t^{(i)}$ as provided. For $t^{(ai)}$ choose one set of possible inputs. Call this as Set 1, and for the later trials Set 2, Set 3, and so on.

F. Make the modified A -construction, as in Fig.6(b), for all unary relations of the A -type. Add the $t^{(am)}$ and $t^{(ao)}$ so produced to the list of terms in A. Add also the Z_j (of type W) that are introduced, to the list in A.

G. Reduce all the multiple paths in unary bouquets to single step unary paths, as in Fig.8. Eliminate the $t_j^{(m)}$ and Z_j so removed from the list in A, and add the new Z_j that are introduced to obtain "reduced" graph.

H. Implement the "reduced graph" using the list of $t^{(i)}$ and $t^{(ai)}$ that are now available, employing only unary and binary forward elementary relations (as in Fig. 6(a) and (c)), and obtain the list of QPL states of $t^{(o)}$ and $t^{(ao)}$. (Rearrange the sequence of the relations as required for this purpose.)

I. Repeat the steps from E to H, for Set 2, Set 3, ... of $t^{(ai)}$ and list the $t^{(o)}$ and $t^{(ao)}$ for each set, as desired.

B) It is assumed that there are no directed circuits in the graph. We check for the occurrence of any such circuits. If there is one, as is shown in Fig. 4(a), it is removed by adding a \vee -operator at one of the terms (say $\underline{\underline{a}}$) for comparing the two newly created terms $\underline{\underline{a}}_1$ and $\underline{\underline{a}}_2$, which lead to $\underline{\underline{a}}_1 \vee \underline{\underline{a}}_2 = \underline{\underline{a}}$ (as in Fig. 4(b)). The term $\underline{\underline{a}}_1$, initiating the (formerly circuitous) path becomes an input term (Type $t^{(i)}$) and $\underline{\underline{a}}_2$ becomes a middle term (Type $t^{(m)}$) while $\underline{\underline{a}}$ will be an output term (Type $t^{(o)}$). This is done until all directed circuits are removed. (We have not yet proved that the graphs obtained by breaking a directed circuit at different terms (like $\underline{\underline{a}}$ above) are equivalent. Probably, they are not. Hence, the point (term $\underline{\underline{a}}$) at which the circuit is cut to provide the "circuit removal construction" must be specified for each circuit before the logic in the graph is algebraically implemented).

C) We then check for the occurrence of any multiple inputs to a term (which may be either of the type $t^{(m)}$ or $t^{(o)}$) and, if such multiple inputs are present for any term, we make the \mathbb{W} -construction, as in Eqs (49) and (50), in each case. The $t^{(m)}$ or $t^{(o)}$ that are thus produced are added to the original list in A, as well as the operators \mathbb{W} that are used. The output of such a \mathbb{W} -operator can be \emptyset for a middle term $t^{(m)}$, and carried forward to later steps of the argument.

D) We then list the reverse binary relations of the type $\mathcal{V}_k(\underline{\underline{c}}_j \underline{\underline{z}}_k \underline{\underline{a}}_j = \underline{\underline{b}}_j)$ and find out if $\underline{\underline{c}}_j$ corresponds to a term which is an original input $t_j^{(i)}$ or an intermediate term $t_j^{(m)}$. If c_j is not a original input, we make the $\underline{\underline{c}}$ -reconstruction as in Fig. 7(b), so that it becomes one of the original inputs. Replace $\mathcal{V}_c(\underline{\underline{c}})$ by $\mathcal{V}_{c1}(\underline{\underline{c1}})$ and introduce the input term $\mathcal{V}_{c2}(\underline{\underline{c2}})$ and the output term $\mathcal{V}_c(\underline{\underline{c}})$. The additional inputs

$t_j^{(ai)}$ and final outputs $t_j^{(ao)}$ thus generated are added to the list in A , but listed separately with the additional superscript (a), as in (ai), (am), (ao). The operator ∇ is added to the list Z_j , as also the Type 1 unary operator $\mathcal{Q}'(\underline{a} Z = \underline{b})$ or $\mathcal{Q}'(\underline{a} Z^c = \underline{b})$ which replaces the binary reverse operator $\mathcal{Q}(c Z a = \underline{b})$. The output of such a ∇ -operator in this process can be \emptyset only for a $t^{(ao)}$, an output term.

E) Taking the set of $t^{(i)}$, $t^{(m)}$ and $t^{(o)}$ that are obtained at this stage, we input the QPL states of those $t_j^{(i)}$ which are provided for the argument. List the $t^{(ai)}$ arising from the c -reconstruction process separately. We choose one set of possible inputs for these (and later modify these over all possible QPL states that could be put into such reconstructed $t_j^{(ai)}$). We consider one such complete set of $t_j^{(i)}$ for implementing the graph of the argument below. When we talk of $t_j^{(i)}$ hereafter, they include both $t^{(ai)}$ and $t^{(i)}$, and similarly for $t_j^{(m)}$ and $t_j^{(o)}$.

F) We now check the unary relations to be of the \tilde{E} -type, \tilde{I} -type and/or \tilde{A} -type. For each of the unary relations of the \tilde{A} -type, we make the modified A-construction as in Fig. 6(b). This also leads to additional $t^{(am)}$ and $t^{(ao)}$, but do not add to the list of the terms $t^{(ai)}$. These are added to the list in A , for implementation, but they will not be changed, as they are the consequences of the process of implementing the unary \tilde{A} -type connectives. A consistency check operator ∇ is also introduced in each case, which could lead to an output \emptyset for a $t^{(ao)}$.

G) Take each of the unary paths in the various unary bouquets that are present in the graph produced at this stage and reduce them to the equivalent single unary relations, as in Figs. 5(a) and (b). Do this for all the unary bouquets that are present. This reduces the number of $t_j^{(m)}$, by eliminating all the

$t_j^{(m)}$ in the bouquets that are neither the starting term, nor the ending term, in each unary path. Remove the Z_j in the original bouquets and introduce the new equivalent Z_j in the single step unary paths.

H) At this stage we obtain a "reduced" graph which contains only (single) unary relations and binary forward relations, and this is the logical graph to be worked out from the $t^{(i)}$ and $t^{(ai)}$, via the Z_j and $t^{(m)}$ to the $t^{(o)}$ and $t^{(ao)}$. Renumber the list of the terms t_j if necessary, and classify them into the types $t^{(i)}$, $t^{(m)}$ and $t^{(o)}$ and write down all the unary and binary forward relations between these terms, which may be in an arbitrary order. (Distinguish between $t^{(i)}$ and $t^{(ai)}$).

Rearrange the sequence of the logical relations thus obtained, so that they can be implemented in serial order. (The

algorithm for this is reserved for a later part of this series.

The proof that this is possible is given in Section 5(a) below).

Using the set of original inputs $t_j^{(i)}$ for the terms $t_j^{(i)}$, implement the equations as newly listed, and obtain the final outputs $t_j^{(o)}$. This can always be done, although some of the middle and/or final outputs may be having the contradictory state \emptyset or the indefinite state Δ .

I) If all the $t_j^{(o)}$ and $t_j^{(ao)}$ are non- \emptyset , then we say that the inputs put in the step E are consistent with the chosen argument, and the conclusions can be read out of these outputs. (See Section 5(b) for a justification of this).

If there are contradictory states among the outputs, choose another set of inputs for the additional inputs $t_j^{(ai)}$, taking care

that the set of original inputs $t_j^{(o)}$ given for the argument

are unchanged, and repeat the process from E to H for

this set of inputs. Once again check for absence of the

contradictory state \emptyset in the outputs $t^{(o)}$. Repeat for all

possible combinations of initial inputs $(t_j^{(ao)})$ other than

those that are given for the argument.

Depending upon the nature of the argument, there may be

only one set of final outputs, or there may be more than one

set of outputs, that are non-contradictory. In the latter case,

the additional inputs, and the intermediate inputs which have

been converted into original inputs in one of the steps, may

have one or the other of different possibilities. We stop the

discussion at this stage, and the analysis of the various types

of arguments, depending on the contradictory and non-contradictory

nature of the outputs is reserved for a future communication.

(a) Proof that a reduced graph consisting of Type-1 statements is implementable.

The proof of this follows the same lines as those adopted in Section 2(e) for the graph of an SNS argument. In fact, the reduced graph for a QPL Type-1 argument is even simpler in structure, since it consists only of unary and binary forward statements. For each of these elementary statements of Type-1, we have already seen that the output term can be worked out in terms of the QPL state of the input for the unary relation, and in terms of the QPL states of the two inputs for a binary statement (see Section 5(c) (i) and (ii)). Although the QPL state of a term t has two parameters representing it, namely the quantifier \forall and the SLS state \underline{t} in the form of $\forall(\underline{t})$, the rules for the determination of the output state of a QPL relation is such that the graphical connection between the input(s) and output follows exactly the same rules as in SNS.

The quantifier(s) of the input(s) and the output, however, require the statement of the relation and the natures of the input quantifiers. However, the rule for this is straightforward and can be written so that they are algorithmically implementable from input to output. Hence for both unary and binary forward relations, statements of Type-1 follow the rules of SIS statements in so far as the structure of the logical graph is concerned and the associated relations between the quantifiers can always be written down for application.

In view of this, we can take over the proof that a graph \mathcal{G}_{n+1} can be implemented for QPL Type-1 statements if we are given that all graphs \mathcal{G}_m ($m = 3$ to n) are implementable. The proof is the same as what is contained in Eqs 22(a,b) for SIS graphs, since all QPL graphs containing a small number (say 10) of vertices are all implementable. In fact, as mentioned above, the proof of this for QPL graphs is unnecessary; because the

structure of the logical graph from the graph theoretical point of view is identical for SNS and QPL, in that they consist of unary and binary relations represented by elementary graphs of the type shown in Fig. 1(a) and (b) for SNS graphs which are taken over to Fig. 6(a) and (c) for QPL Type-1 statements. In view of the isomorphism of the graph-theoretical structure of unary and binary forward statements of SNS and QPL, we shall not give the proof of the above statement, since it is identical with what has been given for SNS graphs in Section 2(e).

As a matter of fact, the graph considered for SNS statements include reverse binary statements, which are avoided in QPL graphs of Type-1 by using the c-reconstruction process, whereby the reverse binary statement is converted into statements of one or the other of the other two types. Hence the QPL Type-1 graph is always implementable using the results given in Figs 6(a) and (b) and Eqs. (30), (32), (33), (34) for the formulae

connecting the terms t_j on the one hand and the quantifiers

\forall_j on the other hand, using the operator Z_k .

b) Processing of the argument through the reduced graph.

As already stated, the reduced graph has only unary and binary forward QPL statements of Type-1. Among the former, $\forall(a \underline{A} = b)$ is not employed in the unary path, but is converted in a suitable way for checking the truth of this statement. Hence, with QPL inputs, the unary paths can only give a QPL output or the state Δ . In the latter, for matrix connectives also, only QPL outputs occur if the inputs are also one of the four QPL states; but if the input is Δ then the output is also Δ .

In the forward direction, \emptyset leads to \emptyset for all connectives (both unary and binary) except for \forall as already mentioned earlier. Hence it follows that the particular set of inputs is contradictory if the \emptyset state occurs anywhere during the working out of the argument. So also Δ leads to Δ for all connectives, except

for \vee , but it does not indicate anything like contradiction.

Instead, this state is of no information value for the resulting

final output having the state Δ . This situation is quite

common, and can be the consequence of an implication, as discussed

in Part III, where such arguments were given the name "invalid".

Actually there is no invalidity in the argument, but only the

result that the conclusion is of no value, just like in the SNS

implication $(\underline{a} \Rightarrow \underline{b}, \neg \underline{a}) \equiv (\underline{b} = D = (1 \ 1))$.

As is obvious, the occurrence of a null set as a result of $\underline{a}_1 \vee \underline{a}_2 = \emptyset$, indicates that the two inputs \underline{a}_1 and \underline{a}_2 into the elementary statement are mutually contradictory and therefore either \underline{a}_1 should be changed to \underline{a}_1^c or an EPL vector contained in it, or \underline{a}_2 must be changed to \underline{a}_2^c or an EPL vector contained in it. Thus, suppose $\underline{a}_1 = \forall \equiv (1 \ 0 \ 0)$ and $\underline{a}_2 = \Delta \equiv (0 \ 1)$ which leads to $\underline{a}_1 \vee \underline{a}_2 = \emptyset$. Then we conclude that \underline{a}_2 should be changed to $(1 \ 0 \ 0) \equiv \forall$, or \underline{a}_1 must be changed to either

(0 1 1), or (0 1 0), or (0 0 1), corresponding to

Λ , Σ or Φ . as the case may be. Beyond this, the treatment

of the consequences of the contradiction will depend upon the

particular argument with regard to retracing of the logic

backwards along the paths in the graph. This is not discussed

here and is reserved for a future communication.

c) Conclusion

In the next Part V, we shall consider QPL arguments having

both Type-1 and Type-2 statements which necessarily also lead

to non-quantified SLS statements. A general logical argument

containing statements belonging both to propositional calculus

and predicate calculus (Type-1 and Type-2 statements) is possible

and leads to no more difficulty in the graph-theoretical

implementation of it than what we have discussed so far. This

is because the flow of logic in the graph is modulated by the

occurrence of terms (t) and unary and binary connectives (Z) in

the same way as we have considered so far. The only new feature that will require consideration is the fact that there will be more than one variable for the different quantifiers that are introduced in the argument. This is being examined and the results ^{will be} ~~✓~~ reported in due course.

FIGURES

	<u>Page No.</u>
1. Elementary statements of S _N S logic	6
2. Logical graph having a tree structure containing unary connectives and binary forward and binary reverse connectives	10
3. A simple logical graph having a closed circuit of edges, in which the outputs from two different paths are combined by the Boolean connective $\underset{\approx}{\vee}$.	16
4. (a) Directed circuit with two unary connectives (b) Inclusion of the operator $\underset{\approx}{\vee}$ and removal of the directed circuit. Note the feed-back line from a to a ₁	21
5. (a) Closed directed circuit with a single term and only one connective (b) Same as (a), but avoiding the directed circuit, but with a feed-back line.	25
6. Elementary connective relations in QPL Type-1 sentences. (a) Unary QPL relation of Type-1 (b) Unary Type-1 relation with (\forall , $\underset{\approx}{A}$) and input $\underset{\approx}{Q}'_a(a')$. The modified A -construction is shown, with outputs $\forall(b')$ and $\underset{\approx}{Q}'_a(a')$. (c) Binary forward QPL relation of Type-1 (d) Binary reverse QPL relation of Type-1	36
7. Schematic diagram of $\underset{\approx}{c}$ -reconstruction (a) Portion of graph before $\underset{\approx}{c}$ -reconstruction representing Eqs.(46a,b) (b) Left: Part of the $\underset{\approx}{c}$ -reconstructed graph with the consistency check operator $\underset{\approx}{W}$. Right: Unary relation between $\underset{\approx}{a}$ and $\underset{\approx}{b}$ generated by $\underset{\approx}{Q}'_{c2}(c2') \equiv \underset{\approx}{Q}_c(c')$.	55

8.	Reduction of a general bouquet of unary paths to a bouquet of reduced single unary relations	
(a)	Bouquet of edges starting from t_1 along unary paths and ending in t_4, t_6, t_7, t_8 and t_9 .	77
(b)	Equivalent single connective unary paths, all starting from t_1 and ending in t_4, t_6, t_7, t_8 and t_9 .	77
9.	Outline of Flow Chart	82

TABLES

Page No.

1. SIS truth values of the terms in Fig.1 for the different T and F inputs for \underline{a} , \underline{b} , \underline{c} .	12
2. List of intermediate terms and output \underline{p} for the logical graph shown in Fig. 3	15
3. Table for the "into" product $\mathcal{V} \otimes \mathcal{V}_a$	38
4. List of the vidya product $\underline{\underline{a}}_1 \otimes \underline{\underline{a}}_2$	60
5. Resultant of two Type-1 implications in succession	72

LIST OF REFERENCES

1. Ramachandran, G.N., Vector-Matrix Representation of Boolean Algebras and Application to Extended Predicate Logic (EPL) Part III (Submitted for publication)
2. -do- Part I, Current Science, 1983, 52, 292-302
3. -do- Part II, Current Science, 1983, 52, 335-341.
4. Ramachandran, G.N., Syad-Nyaya System (SIS)—A New Formulation of Sentential Logic and its Isomorphism with Boolean Algebra of Genus 2, Current Science, 1982, 51, 625-636.

Vector-Matrix Representation of Boolean Algebra and Application
to Extended Predicate logic (EPL)

Part V - Theory for general argument in logic including SNS, QPL
and QSN statements*

G.N. Ramachandran
Gita, 5, 10A Main Road
Malleswaram West
Bangalore 560 055

*Matphil Reports No. 36, June 1984
(Draft 2 - for Private circulation)

Vector-Matrix Representation of Boolean Algebras and Application
to Extended Predicate Logic (EPL)

Part V — Theory for general argument in logic including SNS, QFL
and QSN statements*

G.N. Ramachandran
Gita, 5, 10A Main Road
Malleswaram West
Bangalore 560055

*Matphil Reports No. 36, June 1984
(Draft 2 — for private circulation)

PREFACE AND SUMMARY

A brief account of the framework of this report is given in Section 1 entitled "Introduction". Therefore, only a very brief summary is given here.

In this report it is shown that the method of writing a flow chart for a logical graph consisting of statements in QPL-1 which was given in Part IV, can be completely taken over for the most general type of argument in logic containing terms of the type of SNS, QPL-1 and QPL-2 and relations between any two of these. However, only terms which are functions of one variable are discussed in this report, although relations and the connectives representing the relations may be functions of two or three variables also. While doing this, the algorithm for the working out of a reduced logical graph is given in greater detail than in Part III and is readily seen to be computerizable. Examples of arguments containing various graphical constructions such as c-reconstruction and A-construction as also examples of

.ii.

equations which are in pure SNS or QPL-1 or QPL-2 and also any relations connecting one type with another, are given in the two or three arguments whose logical graphs are given in good detail. It is believed that all the processes described in this report are capable of being converted into steps in a computer program.

As in the case of Part IV, this report has not been checked by anyone other than the author and therefore it cannot be assured that there are no incidental errors or omissions. On the other hand, the main principles and arguments are believed to be correct and properly formulated. Minor lapses may kindly be excused.

<u>CONTENTS</u>	<u>Page No.</u>
1. Introduction	1
2. The four types and three forms of elementary statements	5
(a) SNS relations	7a
(b) QPL-2 relations	10
(c) QSN relations	16
(i) An illustration of the use of QSN relations	20
(ii) Use of $\underline{E}(i, j)$ in QSN - another illustration	23
(d) QPL-1 relations	26
(i) Use of "inverse" relations, in general	30
(ii) Use of inverse relations in QPL-1	33
3. Implementation of the steps of an argument for elementary statements of different types	36
(a) Implementation of binary forward statements	36
(b) Binary reverse and unary statements	37
(c) Implementation of a unary path	39
(d) Review of subsections (a), (b) and (c) and implementation of unary bouquets	44
4. Flow chart for the implementation of a general argument containing all four types of elementary statements	47
(a) Notes and comments on the flow chart	49
(b) Setting up of the flow chart of an argument	57
(c) Rearrangement of the logical equations of a reduced graph to be sequentially implementable	66
(i) Simple example of an argument without circularity	67
(ii) Example with circularity <u>c</u> -reconstruction and <u>A</u> -construction	73

4.(d) Working out the outputs of logical graphs.	
(i) Solution of the reduced graph of the simpler problem	80
(ii) Solution of the argument of the larger graph in Fig. 4(a)	84
(e) Example of an argument containing three types of terms	89
(i) Logical equations of the argument in standard language	92
(ii) Logical graph of the argument and solution of the equations in our language	95
5. Some theorems and special considerations	104
(a) Inversion of an implication considered via consistency checks	104
(b) Other general matrix operators	108
Acknowledgements	112
References	112
List of Figures	113
List of Tables	114

Vector-Matrix Representation of Boolean Algebras and Application
to Extended Predicate Logic (EPL)

Part V — Theory for general argument in logic including SNS, QPL and
QSN statements

1. Introduction

This report contains the extension of the general theory for an argument composed of statements of Type-1 in QPL (hereafter designated as QPL-1), which includes also the theory for an SNS argument, to the case of any general argument which can have statements in SNS (propositional calculus) and in QPL-1 and QPL-2 (which stands for statements of Type-2 in QPL and EPL, which has been discussed extensively in Parts I and II) of predicate calculus. The discussion in this report is made very brief since the proof of almost all the theorems required for the above purpose are contained in the earlier four parts. The essential idea is to show that Section 5 of Part IV in general, and, in particular, the outline of the flow chart of implementing any general argument in QPL-1 (Fig.9 of Part IV), are both readily

extendable to a completely general argument in logic, which can contain statements in SNS, QPL-1, QPL-2 and interconnections between these as will be described below.

The framework of this report may be briefly given here.

In Section 2, the three forms of elementary statements — namely unary, binary forward, and binary reverse — will be described for each of the types SNS, QPL-1, QPL-2 and mixed SNS-QPL or QPL-SI (designated hereafter by QSN). It will be seen from this, that the topological structure of the graph of an argument depends only on the unary and binary (forward or reverse) nature of the relations in it, and is independent of whether it is of one or the other of the above four types. Therefore, in the treatment of the graph in the flow chart, the outline, as given in Fig.9 of Part IV, can be completely taken over. Therefore, in Section 3 of this Part V, each of the steps A to I of this flow chart will be taken and its implementation for each of the above four

types, namely SNS, QPL-1, QPL-2 and QSN will be explained for each of the blocks A to I. As will be seen from this Section, every step of the algorithm in the flow chart is implementable according to well defined rules and formulae for each of the four types of logical relationships. Thereafter, in Section 4, the proof readily follows that any general argument in logic is implementable by using our vector-matrix formalism.

In doing all these, the use of brackets (wherein the fact, that at any stage, a term that is employed is itself the result of an argument consisting of a series of interconnected statements, as indicated by putting them inside a bracket) will not be gone into, except for the recognition of this feature here. This is a well-known principle in mathematics whenever a function of a function is calculated and the same technique can be adopted here also in the algebra representing such a feature. Thus, the statements inside a bracket are taken to form

an independent argument and its graph worked out, by the techniques developed here, to obtain the required term. Then, the output term will be an SNS term, or a QPL term with a quantifier, a variable and an SNS term, in the standard notation for such a QPL term. (The term QPL will be taken to include also EPL terms wherever relevant). Similarly, the way in which different variables are collected and taken care of will not be considered (as this is also a well recognised procedure in mathematics and computer science), except as they occur in the theory of a pair of elementary statements.

The emphasis is to show that a single argument in logic containing a mixture of all possible types of statements that can be put in SNS and QPL, and mixed relations in QSN using both of these, and employing both matrix and Boolean logical connectives, can be implemented, and the outputs calculated, making use of the inputs. In doing this, it will be shown that the rather complicated sequence of operations (Blocks A to I) given in the flow chart in

Fig.9 of Part IV can be followed, and each block replaced by a corresponding one, for each of the four different types of logical relations — namely SNS, QPL-1, QPL-2 and QSN.

2. The four types and three forms of elementary statements.

As mentioned in the previous section, we shall give a brief summary of the three forms of elementary statements — namely unary, binary forward and binary reverse (can have the four types of logic namely SNS, QPL-1, QPL-2 and mixed QSN. The formulae given below will be proved, as all these have been thoroughly discussed in the previous four parts. The reference is made to these parts [1 — 4] and to the earlier publication on SNS logic and theory of relations [5]. Also, as mentioned in the last section, the logical graph indicating the flow of information from term to term is identical for all of these four different types and therefore we shall use a general notation of a, b, c etc., for a term (in general t), and A, O, U, V, etc.,

for a relational connective (Z in general), to represent the terms and operators of the argument in general. t may be thus a SNS term or a QPL term etc., and Z may be a SNS-SNS relation, SNS-QPL-2 relation etc. The structure of the graph for the three elementary logical equations are:

$$\text{Unary} : \underline{a} \underline{Z} = \underline{b} \quad (1)$$

$$\text{Binary forward} : \underline{a} \underline{Z} \underline{b} = \underline{c} \quad (\underline{c} \text{ is an SNS term}) \quad (2)$$

$$\text{Binary reverse} : \underline{c} \overleftarrow{\underline{Z}} \underline{a} = \underline{b} \quad (3a)$$

$$\text{If } \underline{c} = T, \quad \underline{a} \underline{Z} = \underline{b} \quad (3b)$$

$$\text{If } \underline{c} = F, \quad \underline{a} \underline{Z}^C = \underline{b} \quad (3c)$$

$$\text{If } \underline{c} = D, \quad \underline{a} (\underline{Z} \oplus \underline{Z}^C) = \underline{b} \quad (3d)$$

$$\text{If } \underline{c} = X, \quad \underline{a} (\underline{Z} \otimes \underline{Z}^C) = \underline{b} \quad (3e)$$

where \underline{Z}^C stands for the complement of the relation \underline{Z} (see Part I). For the Eqn (3a) the solutions when the state of \underline{c} is given are

outlined in (3b,c,d,e), so that the binary reverse is always

equivalent to a unary relation when the state of \underline{c} is known,

and
so that it is only necessary to give the formula for unary and

binary forward relations, subject to the relevant conditions

for binary reverse in special cases.

The treatment below in this Section 2 gives in outline the relevant formulae for all the four types of logical relations and they in effect give the notation that we are adopting, which for the most part is identical with that used earlier in the previous parts. It is found desirable to use different fonts of letters for the four different types of terms and operators so that an equation itself will suggest the nature of the logic concerned, without having to specifically state that the relevant term is a QPL term, or the relevant relation is an SNS-QPL-2 relation, and so on.

In algebra, an operator \underline{Z} connecting \underline{a} to \underline{b} can be associated with an "inverse" operator (usually designated \underline{Z}^{-1}) connecting \underline{b} to \underline{a} . Thus Eq (1) has the inverse relation $\underline{b} \underline{Z}^{-1} = \underline{a}$. However, we shall use the symbol \underline{Z}^t for this operator, since from the

.7a.

General Theory of Relations discussed in Part I [1], the matrix

$|z|$ connecting the vector $|b\rangle$ to the vector $|a\rangle$ is the

transpose ($|z^t|$) of the matrix $|z|$ connecting the vector $|a\rangle$ to

the vector $|b\rangle$. It then follows that the inverse unary and

binary relations corresponding to Eqs (1), (2), (3) have the form

$$\text{Unary} : \underline{b} \underline{Z}^t = \underline{a} \quad (1')$$

$$\text{Binary forward} : \underline{b} \underline{Z}^t \underline{a} = \underline{c} \quad (2')$$

$$\text{Binary reverse} : \underline{c} \xleftarrow{\underline{Z}^t} \underline{a} = \underline{b} \quad (3'a)$$

$$\text{If } \underline{c} = T, \quad \underline{b} \underline{Z}^t = \underline{a} \quad (3'b)$$

$$\text{If } \underline{c} = F, \quad \underline{b} \underline{Z}^{tc} = \underline{a} \quad (3'c)$$

These inverse relations find particular application in QPL-1, and hence their properties are discussed in Section 2(d)(i) where their application to SNS, QPL-2 and QSN are also discussed.

(a) SNS relations

An SNS term is denoted by $\underline{\underline{t}}$ and an SNS-SNS relational operator denoted by $\underline{\underline{Z}}$, so that the three types of relations take

the form

$$\text{Unary} : \underline{\underline{a}} \underline{\underline{z}} = \underline{\underline{b}} \quad (4a)$$

$$\text{Binary forward} : \underline{\underline{a}} \underline{\underline{z}} \underline{\underline{b}} = \underline{\underline{c}} \quad (4b)$$

$$\text{Binary reverse} : \underline{\underline{c}} \overset{\leftarrow}{\underline{\underline{z}}} \underline{\underline{a}} = \underline{\underline{b}} \quad (4c)$$

As has been discussed in [5], Part I [1] and Part IV [4], the SNS logic, which is the logical extension of propositional calculus, can be represented by Boolean 2-vectors and 2X2 matrices of the type,

$$(t_\alpha \ t_\beta) ; \begin{pmatrix} z_{\alpha\alpha} & z_{\alpha\beta} \\ z_{\beta\alpha} & z_{\beta\beta} \end{pmatrix} \quad (5a, b)$$

Then the formulae giving the output in terms of the input term and the matrix of the relation for the above three types are as follows:

$$\text{Unary} : \langle a | z | = \langle b | \quad (6a)$$

$$\text{Binary forward} : \langle a | z | b \rangle = \langle c | \quad (6b)$$

where

$$\langle a | z | b \rangle = c_\alpha, \quad \langle a | z^c | b \rangle = c_\beta, \quad (c_\alpha \ c_\beta) = \langle c | \quad (6c)$$

This is for matrix operators $\underline{\underline{z}}$.

For Boolean operators $\underline{\underline{U}}$ and $\underline{\underline{V}}$:

$$\langle a | \oplus \langle b | = \langle c | \text{ for } \underline{\underline{U}}, \quad \langle a | \otimes \langle b | = \langle c | \text{ for } \underline{\underline{V}} \quad (6d, e)$$

No formula is given for the binary reverse because by Eqs (3b to e)

it is converted into a unary relation when the SNS state of $\underline{\underline{c}}$

is given. //The four states are T,F,D,X, standing for

$$T = (1 \quad 0) = \text{True} ; \quad F = (0 \quad 1) = \text{False} ; \quad (7a,b)$$

$$D = (1 \quad 1) = \text{Doubtful} ; \quad X = (0 \quad 0) = \text{Impossible} \quad (7c,d)$$

There are 16 possible matrices $|Z|$ corresponding to each of the four components having the Boolean values 1 or 0. Of these, 10 are in general used in current literature in logic; and, in our notation, they are classified as follows:

$$\underline{\underline{E}}, \quad \underline{\underline{N}} \quad - \quad \underline{\underline{E}}\text{-type} \quad (8a)$$

$$\underline{\underline{A}}, \quad \underline{\underline{Q}}^C, \quad \underline{\underline{I}}^C, \quad \underline{\underline{J}}^C \quad - \quad \underline{\underline{A}}\text{-type} \quad (8b)$$

$$\underline{\underline{Q}}, \quad \underline{\underline{A}}^C, \quad \underline{\underline{I}}, \quad \underline{\underline{J}} \quad - \quad \underline{\underline{Q}}\text{-type or } \underline{\underline{I}}\text{-type} \quad (8c)$$

The above operators have, for their matrices, the 2X2 truth table for the logical connective represented by them (see Ref [5]).

In addition, the other two operators found to be of common use are $\underline{\underline{D}}, \underline{\underline{X}}$, given by

$$|\underline{\underline{D}}| = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \quad |\underline{\underline{X}}| = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \quad (8d,e)$$

where $\underline{a} \gtrless \underline{b} = \underline{c}$ leads to $\underline{c} = D$ for all states of \underline{a} and \underline{b} (except X's) and $\underline{a} \gtrless \underline{b} = X$ for all \underline{a} and \underline{b} . The remaining four matrices are not simple connectives, but have properties similar to D and X — for example, \underline{c} is T for all \underline{a} , independent of the state of \underline{b} , and so on. They are representable as sums of elementary matrices of the A-type and are not discussed further. This idea of having a general matrix |Z| as a sum of A-type matrices is of great use for writing complicated logical relations which are not of the standard type, in SNS, QPL and QSN.

They are also representable as a product of two implications, ^{or three} which are simultaneously true. (See Section 5(b)).

(b) QPL-2 relations

These have a structure identical with the SNS relations, except that Boolean 3-vectors and 3X3 Boolean matrices are employed. The notation adopted for relations are as in Eqs. (1), (2) and (3). They are:

$$\text{Unary} \quad : \underline{\underline{a}} \gtrless \underline{\underline{b}} \mapsto \langle \underline{a} | Z | \underline{b} \rangle \quad (9a)$$

Binary forward : $\underline{\underline{z}}^a \underline{\underline{z}}^b = \underline{\underline{c}}$ $\rightarrow \underline{\underline{c}} = (c_\alpha \ c_\beta)$, (9b,c)

where

$$\langle a | z | b \rangle = c_\alpha, \quad \langle a | z^c | b \rangle = c_\beta \quad (9d,e)$$

Binary reverse : $\underline{\underline{c}} \underline{\underline{z}}^a = \underline{\underline{b}}$ (10a)

$$\text{If } \underline{\underline{c}} = T, \quad \underline{\underline{z}}^a = \underline{\underline{b}} \quad (10b)$$

$$\text{If } \underline{\underline{c}} = F, \quad \underline{\underline{z}}^a \underline{\underline{z}}^c = \underline{\underline{b}} \quad (10c)$$

The results for $\underline{\underline{c}} = D$ and X follow from Eqs. (3d) and (3e).

The above are for matrix connectives. For the Boolean connectives \vee and \wedge , the equations follow the pattern of (6d and e), and are

$$\langle a | \oplus \langle b | = \langle c | \text{ for } \vee \quad (11a)$$

$$\langle a | \otimes \langle b | = \langle c | \text{ for } \wedge \quad (11b)$$

The state vectors and matrix connectives of EPL have the forms

$$(t_\gamma \ t_\beta \ t_\alpha) ; \begin{pmatrix} z_{\alpha\alpha} & z_{\alpha\beta} & z_{\alpha\gamma} \\ z_{\beta\alpha} & z_{\beta\beta} & z_{\beta\gamma} \\ z_{\gamma\alpha} & z_{\gamma\beta} & z_{\gamma\gamma} \end{pmatrix} \quad (12a,b)$$

Thus, in addition to the basic states

$$(1 \ 0 \ 0) = \forall = \text{For all} \quad (13a)$$

$$(0 \ 1 \ 0) = \exists = \text{For some, but not for all, nor for none} \quad (13b)$$

$$(0 \ 0 \ 1) = \emptyset = \text{For none} \quad (13c)$$

five more EPL states are recognizable, since there are $2^3 = 8$

possible states for Boolean 3-vectors. These are,

$$(1 \ 1 \ 0) = \exists = \text{There exists} \quad (14a)$$

$$(0 \ 1 \ 1) = \Delta = \text{Not for all} \quad (14b)$$

$$(1 \ 0 \ 1) = \emptyset = \text{For all or none} \quad (14c)$$

$$(1 \ 1 \ 1) = \Delta = \text{Indefinite} \quad (14d)$$

$$(0 \ 0 \ 0) = \emptyset = \text{Impossible (Called "Null Set",}\\ \text{to distinguish it from the}\\ \text{"Impossible" in SNS)} \quad (14e)$$

For ready reference and use, these eight EPL states are designated by $q(1)$ to $q(8)$ and stand for the following:

$$\begin{aligned} q(1) &= (1 \ 0 \ 0), & q(2) &= (0 \ 1 \ 1), & q(3) &= (0 \ 1 \ 0), & q(4) &= (1 \ 0 \ 1), \\ q(5) &= (0 \ 0 \ 1), & q(6) &= (1 \ 1 \ 0), & q(7) &= (1 \ 1 \ 1), & q(8) &= (0 \ 0 \ 0) \end{aligned} \quad (15)$$

They have the property that $q^C(i) = q(i^C)$, where

$$q^C(2j - 1) = q(2j); \quad q^C(2j) = q(2j - 1) \quad (16)$$

These eight 3-vectors are called the "canonical" representations of the logical states and they are extensively used in QPL-2.

The four QPL states alone are represented as follows in the standard literature and they are used as such in QPL-1 (see below).

$$(1 \ 0 \ 0) \equiv (\forall x) (\underline{a}(x)) ; \quad (0 \ 0 \ 1) \equiv (\forall x) (\neg \underline{a}(x)) \quad (17a,b)$$

$$(1 \ 1 \ 0) \equiv (\exists x) (\underline{a}(x)) ; \quad (0 \ 1 \ 1) \equiv (\exists x) (\neg \underline{a}(x)) \quad (17c,d)$$

The representation of QPL-2 connective is a 3×3 Boolean matrix and therefore there can be $512(2^9)$ different matrices.

A way of building all these matrices as sums of only three EPL implications , is indicated in Section 5(b). Here

we shall consider only those matrices which are relevant for use in QPL logical formulae commonly used. Thus, the \approx -type matrix can have 64 different possibilities and they stand for the relations of the type $\forall \& \forall$, $\forall \& \exists$, etc., where the first term can have anyone of the eight states and the second one can have also anyone of the eight states. As shown in Part II [2] the matrix representation for this has the form

$$\approx A(i, j) \mapsto |A(i, j)| = |q(i)\rangle \otimes \langle q(j)| , \quad i, j = 1 \text{ to } 8 \quad (18a)$$

Similarly there are 64 relations of the type $\forall \wedge \forall$, $\forall \wedge \exists$, etc.,

which have for their representation matrices

$$\underset{\approx}{\otimes}(i, j) \mapsto |\underset{\approx}{\otimes}(i, j)| = |\langle q(i) \rangle \oplus \langle q(j) \rangle|, \quad i, j = 1 \text{ to } 8 \quad (18b)$$

In addition to the above, four 3×3 matrices are widely applicable

in QPL-2 logic. These are

$$\underset{\approx}{E} \mapsto \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}; \quad \underset{\approx}{N} \mapsto \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \quad (19a, b)$$

$$\underset{\approx}{D} \mapsto \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \quad \underset{\approx}{X} \mapsto \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad (19c, d)$$

The formulae connecting $\underset{\approx}{A}$ and $\underset{\approx}{Q}^C$, $\underset{\approx}{Q}$ with $\underset{\approx}{A}^C$, $\underset{\approx}{I}$ with 0, $\underset{\approx}{J}$ with $\underset{\approx}{J}$

etc., are not given here and may be obtained from Part II.

However, just as in the case of SNS, it is necessary to recognise

the matrices to be of three types: $\underset{\approx}{E}$ and $\underset{\approx}{N}$ to be of the $\underset{\approx}{E}$ -type,

$\underset{\approx}{A}$, $\underset{\approx}{Q}^C$, $\underset{\approx}{I}^C$, $\underset{\approx}{J}^C$ for all (i, j) to be of the $\underset{\approx}{A}$ -type and $\underset{\approx}{Q}$, $\underset{\approx}{A}^C$, $\underset{\approx}{I}$, $\underset{\approx}{J}$,

for all (i, j) to be of the $\underset{\approx}{Q}$ (or $\underset{\approx}{I}$)-type.

As is seen from Eqs. (9) and (10), the algebra of the 3×3 matrices in QPL-2 closely follow the pattern of the 2×2 matrices in SNS; only they are more in number and the equations contain more parameters. These 3×3 matrices relate one or two QPL terms to another QPL term and, in general, QPL relation in terms of quantifier variables has the following form for unary and binary relations (20a and b), and for one example of binary reverse relation (20c):

$$\underline{\underline{a}}(x) \underset{\approx}{\sim} (x, y) = \underline{\underline{b}}(y) \quad (20a)$$

and

$$\underline{\underline{a}}(x) \underset{\approx}{\sim} (x, y) \underline{\underline{b}}(y) = \underline{\underline{c}} \quad (20b)$$

$$\underline{\underline{c}} = T, \underline{\underline{c}} \overset{\leftarrow}{\approx} (x, y) \underline{\underline{a}}(x) = \underline{\underline{b}}(y) \mapsto \underline{\underline{a}}(x) \underset{\approx}{\sim} (x, y) = \underline{\underline{b}}(y) \quad (20c)$$

These forms are the most general ones that can happen; but it is also possible to have relations between terms having less number of variables or having the same variable throughout.

This can happen, for example, by having $x = y$ in (20a), (20b) or (20c). These are particular cases of the general result and

do not require any special attention. In what is discussed both so far, and later, with reference to QPL-2, we shall use only the general structure ^{for variables} ~~as given for Eqs (20a to c).~~

(c) QSN relations

We shall consider the mixed SNS-QPL and QPL-SNS types of relations before considering ^{the} QPL-1 type of relation, because these have an algebraic structure closely resembling what we have discussed already in the subsections (a) and (b) for SNS-SNS and QPL-2 - QPL-2 relations.

The mixed relations QSN, of the types SNS-QPL and QPL-SNS are closely similar and therefore they are considered together. Both have the same form for unary, binary forward and binary reverse relations as in Eq. (6) for SNS and in Eqs (9) and (10) for QPL-2. In the corresponding logical equations for mixed relations, we indicate SNS vectors by \underline{a} , \underline{b} etc., and QPL vectors in the canonical form by $\underline{\underline{a}}$, $\underline{\underline{b}}$ etc. To distinguish the inter-connecting matrix to be ^{of} the mixed type which may be a 2×3 matrix

for SNS-QPL or a 3×2 matrix for QPL-SNS, we designate it by $\underline{\underline{Z}}$.

Thus, for the mixed case, the three standard forms of elementary relations are as follows:

SNS-QPL

$$\text{Unary} : \underline{\underline{a}} \underline{\underline{Z}} = \underline{\underline{b}} \quad (21a)$$

$$\text{Binary forward} : \underline{\underline{a}} \underline{\underline{Z}} \underline{\underline{b}} = \underline{\underline{c}} \quad (21b)$$

$$\text{Binary reverse} : \underline{\underline{c}} \overset{\leftarrow}{\underline{\underline{Z}}} \underline{\underline{a}} = \underline{\underline{b}} \quad (21c)$$

QPL-SNS

$$\text{Unary} : \underline{\underline{a}} \underline{\underline{Z}} = \underline{\underline{b}} \quad (22a)$$

$$\text{Binary forward} : \underline{\underline{a}} \underline{\underline{Z}} \underline{\underline{b}} = \underline{\underline{c}} \quad (22b)$$

$$\text{Binary reverse} : \underline{\underline{c}} \overset{\leftarrow}{\underline{\underline{Z}}} \underline{\underline{a}} = \underline{\underline{b}} \quad (22c)$$

The corresponding vectors for $\underline{\underline{a}}$ and $\underline{\underline{a}}$ are $(a_\alpha \ a_\beta)$ and $(a_\gamma \ a_\delta \ a_\epsilon)$.

Only the matrix for the connective operator has a new form and

it is as follows:

$$\text{SNS-QPL: } \underline{\underline{Z}} \mapsto 2 \times 3 \text{ matrix} \begin{pmatrix} z_{\alpha\gamma} & z_{\alpha\delta} & z_{\alpha\epsilon} \\ z_{\beta\gamma} & z_{\beta\delta} & z_{\beta\epsilon} \end{pmatrix} \quad (23a)$$

$$\text{QPL-SNS: } \underline{\underline{Z}} \mapsto 3 \times 2 \text{ matrix} \begin{pmatrix} z_{\gamma\alpha} & z_{\gamma\beta} \\ z_{\delta\alpha} & z_{\delta\beta} \\ z_{\epsilon\alpha} & z_{\epsilon\beta} \end{pmatrix} \quad (23b)$$

The matrices of the A-type and O-type can be constructed in the same manner as in Eqs. (18a,b) for QPL-2. For this purpose, the four SNS states can be designated as in (24a):

$$s(1) = (1 \ 0), \quad s(2) = (0 \ 1), \quad s(3) = (1 \ 1), \quad s(4) = (0 \ 0) \quad (24a)$$

with formulae for complementing the vectors similar to those in (16), namely

$$s^c(2j - 1) = s(2j); \quad s^c(2j) = s(2j - 1) \quad (24b)$$

Then, we have, for SNS-QPL matrices of the A and O-types, the formulae (25), in which the SNS index is underlined:

$$\underline{A}(i, j) \mapsto |A(\underline{i}, j)\rangle = |s(i)\rangle \otimes \langle q(j)| \quad (25a)$$

$$\underline{O}(i, j) \mapsto |O(\underline{i}, j)\rangle = |s(i)\rangle \oplus \langle q(j)| \quad (25b)$$

For the QPL-SNS matrices of the same two types, we have similarly

$$\underline{\underline{A}}'(i, j) \mapsto |A'(i, \underline{j})\rangle = |q(i)\rangle \otimes \langle s(j)| \quad (26a)$$

$$\underline{\underline{O}}'(i, j) \mapsto |O'(i, \underline{j})\rangle = |q(i)\rangle \oplus \langle s(j)|$$

It follows from the definitions in (25) and (26) that

$$|A'(i, j)| = |A^t(j, i)| ; \quad |O'(i, j)| = |O^t(j, i)| \quad (27a, b)$$

so that we may write for the operators also, the symbolic equations

$$\underline{A}'(i, j) = \underline{A}^t(j, i) ; \quad \underline{O}'(i, j) = \underline{O}^t(j, i) \quad (28a, b)$$

and it is sufficient to define the matrices of the \underline{A} -type and \underline{O} -type for the SNS-QPL connectives, and their transposes give the corresponding matrices for QPL-SNS relations.

The $\underline{\underline{E}}$ -type connectives are not relevant for 2×3 and 3×2 matrices. However, the Boolean binary operators U and V are relevant. For using these, the 2-vector $(s_\alpha \ s_\beta)$ can be extended to form a three-vector $(s_\gamma' \ s_\delta' \ s_\epsilon')$ by the following rules:

$$\underline{s} = (1 \ 0) \mapsto \underline{\underline{s}}' = (1 \ 0 \ 0), \quad \underline{s} = (0 \ 1) \mapsto \underline{\underline{s}}' = (0 \ 0 \ 1) \quad (29a, b)$$

$$\underline{s} = (1 \ 1) \mapsto \underline{\underline{s}}' = (1 \ 1 \ 1), \quad \underline{s} = (0 \ 0) \mapsto \underline{\underline{s}}' = (0 \ 0 \ 0) \quad (29c, d)$$

It is readily verified that they are logically reasonable. Then for the logical operators "upon" and "vidya" we use the following

equations,

$$\underline{s} \quad \underline{U} \quad \underline{q} = \underline{q} \quad \underline{U} \quad \underline{s} \rightarrow \underline{\text{ns}}' \quad \underline{\text{U}} \quad \underline{q} \rightarrow \langle s' | \oplus \langle q | \quad (30)$$

$$\underline{s} \quad \underline{V} \quad \underline{q} = \underline{q} \quad \underline{V} \quad \underline{s} \rightarrow \underline{\text{ns}}' \quad \underline{\text{V}} \quad \underline{q} \rightarrow \langle s' | \otimes \langle q | \quad (31)$$

For the binary reverses of the QSN type in (21c) and (22c),

the following hold (with algebra identical with SNS and QPL-2 statements) as in (32) and (33) below:

$$(21c) : \underline{c} = T \rightarrow \underline{a} \underline{Z} = \underline{b} ; \quad \underline{c} = F \rightarrow \underline{a} \underline{Z}^C = \underline{b} \quad (32a,b)$$

$$(22c) : \underline{c} = T \rightarrow \underline{a} \underline{Z} = \underline{b} ; \quad \underline{c} = F \rightarrow \underline{a} \underline{Z}^C = \underline{b} \quad (33a,b)$$

(i) An illustration of the use of QSN relations

The following simple statement connecting three QPL terms, as in a binary relation, requires the use of an intermediate SNS term and a QSN relation of the type SNS-QPL.

"If all categories of food are available and some beverages are also available, then all persons present are not unhappy" (34)

We use x as variable and \underline{f} to represent "food available"; y as

variable and \underline{b} to represent "beverage available" and z as variable for persons and \underline{u} for "unhappy". Then (34) can be symbolized as follows:

$$(\forall x) (\underline{f}(x)) \& (\exists y) (\underline{b}(y)) \Rightarrow (\forall z) (\neg \underline{u}(z)) \quad (35)$$

However, to work this out by the vector-matrix representation, we have to use an intermediate SNS term (\underline{g} say) and write it as follows:

$$(\forall x) (\underline{f}(x)) \& (\exists y) (\underline{b}(y)) = \underline{g} \quad (\text{QPL-2 binary forward}) \quad (36a)$$

$$\underline{g} \Rightarrow (\Phi z) (\neg \underline{u}(z)) \quad (\text{SNS-QPL unary}) \quad (36b)$$

Just to make the story complete, we shall give below the matrices to be employed in (36a) and (36b):

$$(36a) : \underline{z}_1 \mapsto |z_1| = |\underline{q}(1) \otimes \underline{q}(6)| = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad (37a)$$

$$(36b) : \underline{z}_2 \mapsto |z_2| = |\underline{s}(2) \oplus \underline{q}(5)| = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix} \quad (37b)$$

An interesting variation of the practical statement in (34) can be written, having only a single variable, as follows (see comments following Eqs. (20a,b,c)).

"If all persons have food and some have wine, then all are happy" (38a)

The relevant logical equation, similar to (35), using an obvious notation, is the following.

$$(\forall z) (\underline{f}(z)) \& (\exists z) (\underline{w}(z)) \Rightarrow (\forall z) (\underline{h}(z)) \quad (38b)$$

and this also leads to two equations as in (36), which need two matrices, one 3×3 and the other 2×3 , for their practical implementation.

The generalization of this to $\mathcal{V}_a(\underline{a}) \approx \mathcal{V}_b(\underline{b}) \Rightarrow \mathcal{V}_c(\underline{c})$ is obvious, and it can be worked out, using two equations

$$\mathcal{V}_a(\underline{a}) \approx \mathcal{V}_b(\underline{b}) \equiv \underline{\underline{a}} \approx \underline{\underline{b}} = \underline{\underline{g}} \quad (39a)$$

$$\underline{\underline{g}} \cdot \underline{\underline{I}}(1, 1) = \underline{\underline{c}} \equiv \mathcal{V}_c(\underline{c}) \quad (39b)$$

See Eq.(46) below for the matrix representation of $\underline{\underline{a}} \approx \underline{\underline{b}} \equiv \underline{\underline{c}}$.

.23.

(ii) Use of $\underline{\underline{E}}(\underline{i}, \underline{j})$ in QSN - another illustration

Although we had said earlier that the diagonal QPL matrices $|E|$ and $|N|$ have no counterpart in QSN, since the QSN matrices are rectangular (being 2×3 or 3×2), equivalence relations between SNS terms and QPL terms can be written, and represented via QSN matrices $\underline{\underline{E}}(\underline{i}, \underline{j})$, as is shown below.

We first take an example, for $\underline{\underline{a}} \underline{\underline{Z}} = \underline{\underline{b}}$ as in (40).

If $\underline{\underline{a}} = T$, then $\underline{\underline{b}} = \forall$; if $\underline{\underline{a}} = F$, then $\underline{\underline{b}} = \exists \forall = \wedge$ (40)

In order to write the matrices, we first need the matrix for $\underline{\underline{I}}(\underline{i}, \underline{j})$ and $\underline{\underline{I'}}(\underline{i}, \underline{j})$, which are readily written down from the logical formulae

$$\underline{\underline{a}} \Rightarrow \underline{\underline{b}} \equiv \neg \underline{\underline{a}} \vee \underline{\underline{b}} ; \quad \underline{\underline{a}} \Rightarrow \underline{\underline{b}} \equiv \neg \neg \underline{\underline{a}} \vee \underline{\underline{b}} \quad (41a,$$

Hence, the matrices $|I|$ and $|I'|$ can be written in terms of $|O|$ and $|O'|$ as

$$|I(\underline{i}, \underline{j})| = |O(\underline{i}^c, \underline{j})| ; \quad |I'(\underline{i}, \underline{j})| = |O'(\underline{i}^c, \underline{j})| \quad (42a,$$

Now, considering the relation (40), the following are simultaneously true:

$$\langle a | I(\underline{1}, 1) | = \langle b | ; \quad \langle a | I(\underline{2}, 2) | = \langle b | \quad (43a,b)$$

Hence, using the principle of the vidya operator, the matrix that will represent the two combined relations in (40) is $E(\underline{1}, 1)$ in the relation

$$\langle a | E(\underline{1}, 1) | = \langle b | , \text{ where } |E(\underline{1}, 1)| = |I(\underline{1}, 1)| \otimes |I(\underline{2}, 2)| \quad (44a,b)$$

and Hence,

$$|E(\underline{1}, 1)| = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix} \quad (44c)$$

It is readily verified that the matrix so obtained in (44c) has the required properties.

More generally, the combined relation

$$\underline{\underline{a}} (= s(i)) \Rightarrow \underline{\underline{b}} (= q(j)) ; \quad \neg \underline{\underline{a}} (= s(i^c)) \Rightarrow \neg \underline{\underline{b}} (= q(j^c)) \quad (45a,b)$$

has the form $\langle a | E(\underline{i}, j) | = \langle b |$, with

$$|E(\underline{i}, j)| = |I(\underline{i}, j)| \otimes |I(\underline{i}, j^c)| \quad (45c)$$

The formula (45) is extremely useful in dealing with QPL

relations of the type $\underline{\underline{a}} \underline{\underline{Z}} \underline{\underline{b}} \equiv \underline{\underline{c}}$, for this equation is

implementable via the following matrices

$$\langle a | z | b \rangle = g_\alpha; \quad \langle a | z^c | b \rangle = g_\beta; \quad \langle g | E | = \langle c | \quad (46a)$$

where $|E|$ is a 2×5 matrix of the form (45c). The details are omitted.

So also, the corresponding formulae for QPL-SNS relations of the type $\underline{\underline{a}} \underline{\underline{E}}'(i, j) = \underline{\underline{b}}$ are obvious, and are not considered in detail. It may be mentioned that more specialised relations of the form as in (40) can be written down. For example, consider the relation

$$\text{If } \underline{\underline{a}}, \text{ then } \underline{\underline{b}} = \forall; \text{ if } \neg \underline{\underline{a}}, \text{ then } \underline{\underline{b}} = \emptyset \quad (47a)$$

This can be represented by $\underline{\underline{a}} \underline{\underline{Z}} = \underline{\underline{b}}$, with the QSN-matrix

$$|z| = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (47b)$$

This was, in fact, used in the problem discussed in Section 7(c) of Part II [2]. The general principles of examples of this type are briefly discussed later in sections 4 and 5, including the analog of the QSN-equivalence operator $E(i, j)$ in QPL, namely $\underline{\underline{E}}(i, j)$.

(d) QPL-1 relations

Since QPL-1 algebra has been extensively discussed in Parts III and IV [5, 4], only the essentials will be mentioned here, so as to correlate it to QPL-2, SNS and QSN. To distinguish them from QPL-2 relations, the QPL terms in QPL-1 relations are indicated by the symbols \mathcal{Q} , \mathcal{B} , etc., and the relational operators are indicated by \mathcal{Z} , \mathcal{A} , \mathcal{O} etc. In terms of these, the "formal" relations can be expressed in the form of Eqns (1), (2), (3), \mathcal{t} replacing \underline{t} and \mathcal{Z} replacing \underline{z} , as in QPL-2; but, for practical application, the relations obey an algebra that is quite different.

Thus, a term \mathcal{Q} is written in terms of the quantifier as $(\mathcal{V}x) (\underline{a}(x))$, simplified to $\mathcal{V}(\underline{a})$. We shall call it the standard form, to distinguish it from the canonical form $\underline{\underline{a}}$ of \mathcal{Q} . The two forms are interconvertible, using the canonizer for going from \mathcal{Q} to \underline{a} , and the standardizer for going from $\underline{\underline{a}}$ to \mathcal{Q} .

(The standard form used in the general theory of this Part. V, following the treatment in Part III, is slightly different from that used in Part II; but there will be no difficulty in redefining the standardizer in Table 7 of Part II, using the notation of Parts III and IV. The essential difference is that the four standard states (\emptyset) of QPL are taken to be those in (17a,b,c,d), and their equivalent canonical forms $\underline{\underline{a}}$ are also given in (17)).

In the unary and binary relations that are special to QPL-1, the relational operator \mathcal{Z} has two parts \mathcal{Q} and \mathcal{Z} and may be represented formally as $\mathcal{Z} = (\mathcal{Q}, \mathcal{Z})$. Then, we give below the format of the three forms of relations as in Eqs (1), (2), (3):

Unary

$$\mathcal{V}_a(\underline{\underline{a}}'), \quad \mathcal{V}_{\underline{\underline{a}} \mathcal{Z} = \underline{\underline{b}}} \mapsto \mathcal{V}_b(\underline{\underline{b}}') \quad (48a)$$

has the solution

$$\underline{\underline{a}}' \mathcal{Z} = \underline{\underline{b}}', \quad (\mathcal{V} \otimes \mathcal{V}_a) \otimes \mathcal{V}_{\mathcal{Z}} = \mathcal{V}_b \quad (48b,c)$$

The new Boolean connective "into" (\oplus) is defined in Section 3(b) of Part IV, and \mathcal{V}_z is also defined therein. However, for ready reference, the table for "into", as operative in QPL-1, is given below in (49). The operation \oplus is both commutative and associative.

$$\left. \begin{array}{l} V \oplus V = V; \quad V \oplus \exists = \exists; \quad V \oplus \Delta = \Delta; \\ \exists \oplus \exists = \Delta; \quad \exists \oplus \Delta = \Delta; \quad \Delta \oplus \Delta = \Delta. \end{array} \right\} \quad (49)$$

Binary forward

$$\mathcal{V}_a(\underline{\underline{a}}'), \mathcal{V}_b(\underline{\underline{b}}'), \mathcal{V}(\underline{\underline{a}} \gtrless \underline{\underline{b}} = \underline{\underline{c}}) \mapsto \mathcal{V}_c(\underline{\underline{c}}') \quad (50a)$$

has the solution

$$\underline{\underline{a}}' \gtrless \underline{\underline{b}}' = \underline{\underline{c}}', \quad \text{and} \quad f(\mathcal{V}_a, \mathcal{V}_b, \mathcal{V}_c, \gtrless) = \mathcal{V}_c \quad (50b,c)$$

(See Part IV for the nature of the function f in (50c).) The

above is for a matrix connective \gtrless in $\mathbb{Z} = (\mathcal{V}, \gtrless)$. For $\mathbb{Z} = \text{U}$ and W , see below, after (51).

Binary reverse

As given in Part IV, this has the formulae:

$$\mathcal{V}_c(\underline{\underline{c}}'), \mathcal{V}_a(\underline{\underline{a}}'), \mathcal{V}(\underline{\underline{c}} \overset{\leftarrow}{\gtrless} \underline{\underline{a}} = \underline{\underline{b}}) \mapsto \mathcal{V}_b(\underline{\underline{b}}')$$

where

$$\underline{\underline{a}}' \gtrless \underline{\underline{b}}' \quad \text{if } \underline{\underline{c}}' = T; \quad \underline{\underline{a}}' \overset{\leftarrow}{\gtrless} \underline{\underline{b}}' \quad \text{if } \underline{\underline{c}}' = F \quad (51a,b)$$

and

$$(\mathcal{V} \otimes \mathcal{V}_c \otimes \mathcal{V}_a) \oplus (\mathcal{V}_z \text{ or } \mathcal{V}_z c) = \mathcal{V}_b, \text{ for } \underline{c} = T \text{ or } F \quad (51c)$$

However, in the spirit of Eq.(3) of converting the binary reverse into a unary relation, we may rewrite (51) below, as in (52).

$$\mathcal{V}_c(\underline{c}'), \mathcal{V}(\underline{c} \xrightarrow{\leftarrow} \underline{a} = \underline{b}) \mapsto \mathcal{V}'(\underline{a} \xrightarrow{\sim} \underline{c}), \text{ if } \underline{c} = T; \quad (52a)$$

$$\mapsto \mathcal{V}'(\underline{a} \xrightarrow{\sim} \underline{c}) \text{ if } \underline{c} = F; \quad (52b)$$

and, in both cases,

$$\mathcal{V} \otimes \mathcal{V}_c = \mathcal{V}' \quad (52c)$$

The formulae for $\mathbb{Z} = \mathbb{U}$ and \mathbb{V} are identical with Eqs.(11a,b) for QPL-2 relations, and the relations take the form

$$\mathcal{A} \mathbb{U} \mathcal{B} = \underline{a} \oplus \underline{b}; \quad \mathcal{A} \mathbb{V} \mathcal{B} = \underline{a} \otimes \underline{b} \quad (53a, b)$$

Thus, if a relation \mathbb{U} or \mathbb{V} is encountered in a QPL-1 argument, then the QPL-1 input terms involved in it are converted into their canonical forms and (53a or b) is applied to get the output in the canonical form, which is then converted into the standard

form, for further QPL-1 manipulations. To avoid these complicated manipulations, short tables for $\text{U}(\oplus)$ and $\text{V}(\otimes)$ for QPL-1 terms (analogous to (49) for \otimes) are given below. (Both are commutative and associative).

$$\left. \begin{array}{l} A \oplus A = A; A \oplus E = E; A \oplus \Delta = \Delta \\ E \oplus E = E; E \oplus \Delta = \Delta; \Delta \oplus \Delta = \Delta \end{array} \right\} \quad (54a)$$

$$\left. \begin{array}{l} A \otimes A = A; A \otimes E = A; A \otimes \Delta = A \\ E \otimes E = E; E \otimes \Delta = E; \Delta \otimes \Delta = \Delta \end{array} \right\} \quad (54b)$$

So also, if a QPL-2 output is used as QPL-1 input, or vice versa, the change-over occurs only at a term (t), and hence it can be carried out using the transformation of the standardizer or canonizer for QPL terms.

(i) Use of "inverse" relations, in general

As mentioned in the beginning of this Section 2, equivalent inverse relations (1'), (2'), (3') can be written for the corresponding

original relations (1), (2), (3). In each case, the relation

from a to b is inverted so that it is from b to a. This finds

the greatest application for the compound ^{disj}~~compon~~ of two QPL-1 relations

in sequence; but its beauty is best appreciated in SNS (and in

QPL-2 and QSN). In all the ^{latter three} cases, the vector-matrix formulae

that occur are identical, and we shall first write them for

SNS elementary relations. Thus Eqs (1) and (1') have their representations as in (55a,b), and they are equivalent (see Theory of Relations in Part I, Section 2(a) — the word "reverse" used therein is to be replaced by "inverse").

$$(\langle a | z | = \langle b |) \equiv (\langle b | z^t | = \langle a |) \quad (55a,b)$$

As a simple example, we have

$$(\underline{a} \Rightarrow \underline{b}) \equiv (\neg \underline{b} \Rightarrow \neg \underline{a}) \quad (56a)$$

$$\text{or } (\underline{a} \underset{\sim}{\sim} \underline{b}) \equiv (\underline{b} \underset{\sim}{\sim} \underline{a}), \quad \underline{J} = \underline{I}^t \equiv \underline{N} \underset{\sim}{\sim} \underline{N} \quad (56b,c)$$

The counterparts for (2) and (2') are, for this example,

$$(\underline{a} \underset{\sim}{\sim} \underline{b}) \equiv (\underline{b} \underset{\sim}{\sim} \underline{a}) = \underline{c} \quad (57)$$

whose matrix representations are

$$\langle a | I | b \rangle = c_\alpha, \quad \langle a | I^c | b \rangle = c_\beta \quad (58a)$$

$$\langle b | I^t | a \rangle = \langle b | J | a \rangle = c_\alpha, \quad \langle b | I^{tc} | a \rangle = \langle b | J^c | a \rangle = c_\beta \quad (58b)$$

The generalization of this follows from the identities in the

Theory of Relations [1] ^(and not w/) given in (59a,b), namely

$$\langle a | z | b \rangle \equiv \langle \phi | z^t | a \rangle; \quad \langle a | z^c | b \rangle \equiv \langle b | z^{tc} | a \rangle \quad (59a,b)$$

The inter-relation between (3a) and (3'a) is similar in that

they go over into the forms of the l.h.s and r.h.s. of (55a,b).

Hence, these are not discussed in detail.

The most interesting consequence of the Boolean vector-matrix representation is that the equivalences in (55) and in (59a,b) can be completely taken over for QPL-2 and QSN also. The only differences are that the nature of the matrices is different in each case, as in (60):

$$|z| \text{ is } 2 \times 2; \quad |z^t| \text{ is } 2 \times 2 \quad \text{for SNS-SNS} \quad (60a)$$

$$|z| \text{ is } 3 \times 3; \quad |z^t| \text{ is } 3 \times 3 \quad \text{for QPL-QPL} \quad (60b)$$

$|z|$ is 2×3 ; $|z^t|$ is 3×2 for SNS-QPL (60c)

$|z|$ is 3×2 ; $|z^t|$ is 2×3 for QPL-SNS (60d)

An important consequence is that, for all these four types of relations,

$$(|z_1| z_2 | \dots |z_k|)^t = |z_k^t| \dots |z_2^t| z_1^t | \quad (61)$$

leading to the fact that two unary relations can be combined either in the direct or in the inverse forms; and they lead to equivalent resultants (see Section 3 below).

(ii) Use of inverse relations in QPL-1

In general, the formula giving the equivalences of inverse relations in QPL-1 is the same as mentioned above, namely,

$$\mathcal{Q}/(\underline{a} \underline{Z} = \underline{b}) \equiv \mathcal{Q}/(\underline{b} \underline{Z}^t = \underline{a}), \text{ for } \underline{Z} = A\text{-type, } Q\text{-type and } E\text{-type} \quad (62)$$

However, there is an exception for \underline{I} -type relations inside the bracket that is quantified. This feature has been discussed in great detail in Section 3(c)(ii) of Part III [3], and so only

the results are given here. Thus, for \mathcal{I} -type relations,

namely with $\mathcal{Z} = \mathcal{N}$, $\mathcal{N} \mathcal{I} \mathcal{L}$, $\mathcal{L} \mathcal{N}$, $\mathcal{N} \mathcal{I} \mathcal{N} (= \mathcal{J})$,

$$\forall(\underline{\underline{a}} \mathcal{I} = \underline{\underline{b}}) \equiv \forall(\neg \underline{\underline{b}} \mathcal{I} = \neg \underline{\underline{a}}) \quad (63a)$$

but

$$\exists(\underline{\underline{a}} \mathcal{I} = \underline{\underline{b}}) \equiv \exists(\underline{\underline{b}} \mathcal{I} = \underline{\underline{a}}) \quad (63b)$$

The latter arises from the well-known relation "For some, if

$\underline{\underline{a}}$ then $\underline{\underline{b}}$ " is to be symbolized as $(\exists x)(\underline{\underline{a}}(x) \& \underline{\underline{b}}(x))$, (see

for example p. 50 of the book [6] by Suppes). Now, since

$$\underline{\underline{a}} \& \underline{\underline{b}} \equiv \underline{\underline{b}} \& \underline{\underline{a}},$$

$$(\exists x)(\underline{\underline{a}}(x) \& \underline{\underline{b}}(x)) \equiv (\exists x)(\underline{\underline{b}}(x) \& \underline{\underline{a}}(x)) \quad (64a)$$

Hence

$$\text{"For some, if } \underline{\underline{a}} \text{ then } \underline{\underline{b}}" \equiv \text{"For some if } \underline{\underline{b}}, \text{ then } \underline{\underline{a}}" \quad (64b)$$

In fact, Eq (64b) can be deduced also straightforwardly from the Venn

diagram showing the relation between the sets $\underline{\underline{a}}(x)$ and $\underline{\underline{b}}(x)$, as

shown in Fig.2 of Part III. Hence, symbolically, we can write

(64b) in the form of (63b), and we obtain the important fact that

the inverting of the relation from $\underline{\underline{a}}$ to $\underline{\underline{b}}$ in (63a) and (63b) have

different algebraic forms. In particular, (63a) obeys the general rule (62), since $\underline{N} \underline{I} \underline{N}$ which occurs on the r.h.s. of it is the same as $\underline{J} \equiv \underline{I}^t$. However, (63b) is quite different. Consequently, the compounding of two QPL-1 implications requires new rules, as discussed in great detail in Part III and briefly again in Section 4(f)(i) of Part IV. The essentially new result that emerges is that if two unary implications in QPL-1 have quantifiers \exists and \forall , or \forall and \exists , respectively, they can be combined to lead to a single implication, as in (65) below (see also Table 5 of Part IV):

$$\exists(\underline{a} \underline{I} = \underline{b}), \quad \forall(\underline{b} \underline{I} = \underline{c}) \equiv \exists(\underline{a} \underline{I} = \underline{c}) \quad (65a)$$

$$\begin{aligned} & \forall(\underline{a} \underline{I} = \neg \underline{b}), \quad \exists(\underline{b} \underline{I} = \underline{c}) \\ & \equiv \text{Inverse of } (\exists(\underline{c} \underline{I} = \underline{b}), \quad \forall(\underline{b} \underline{I} = \neg \underline{a})) \\ & \equiv \text{Inverse of } \exists(\underline{c} \underline{I} = \neg \underline{a}) \equiv \exists(\neg \underline{a} \underline{I} = \underline{c}) \quad (65b) \end{aligned}$$

These are the categorical syllogisms Darii and Dimension, in mood 1, considered in Part III, of which the latter is not contained in the classical list of 18 valid syllogisms, but is one of the new ones derived by us in that report.

3. Implementation of the steps of an argument for elementary statements of different types.

We shall show in Section 4 that the different steps of the Flow Chart in Fig.9 of Part IV [4] (for arguments of QPL-1) are implementable for SNS, QPL-1, QPL-2 and QSN statements. In order to facilitate the understanding of its capabilities, we shall consider in this Section 3, the main features of elementary statements, a pair of unary statements, of a chain of such statements forming a unary path, and of a unary bouquet, for all these four types.

(a) Implementation of binary forward statements

A binary statement connecting a and b with c, namely

$$\underline{a} \underset{\text{Z}}{\sim} \underline{b} = \underline{c} \quad (66)$$

is always implementable given the inputs a' and b', for all four types SNS, QPL-2, QSN and QPL-1. For the first three, the algebra is identical, as in (6) and (9) carried over to QSN, as

mentioned in Section 2(c). It follows that, in (66), the term

a can be any of the three types and b also can be one of the three types, and it can be implemented, using the right mxn matrix,

For QPL-1 also, given $\underline{a} \equiv Q/(a)$ and $\underline{b} \equiv Q/(b)$ (note that QPL-1 and QPL-2 forms are interchangeable) and the relation $Q/(a \gtrless b)$, the state of c in Eq.(66) can be worked out uniquely, as in Section 3(c)(i) of Part IV . Hence, for all possible types, the output c of a binary forward statement can be worked out, when we are given the inputs.

(b) Binary reverse and unary statements

The binary reverse form always leads to a unary relation, as shown in Eqs.3(a) to (e), for all four types, and this creates no problem, since the forward relations from which they are derived are very similar. So also, for a single unary statement of a general type, given the relation $\underline{a} Z = \underline{b}$ and the input \underline{a}' , the output \underline{b}' in $\underline{a}' Z = \underline{b}'$ is calculable from straightforward

rules. For SNS-SNS, QPL-2-QPL-2, SNS-QPL-2 and QPL-2-SNS

relations, the matrix relations have the same form, $\langle a' | z | = \langle b' |$,

and create no problem. If the relation is of the QPL-1 type,

$\mathcal{V}(a \underline{z} = b)$, then Eq.(48) is applicable, and here again, the output $\mathcal{V}_b(b')$ follows, when we are given $\mathcal{V}_a(a')$ and the relation

$\mathcal{V}(a \underline{z} = b)$. This is particularly so when \underline{z} is an I-type or

E-type connective operator. However when \underline{z} is one of the four

A-type connectives in QPL-1, we have to go back to the original

relation $\mathcal{V}(a \underline{z} b = T)$ from which it was derived, and the

A-construction (as in Fig.6(b) of Part IV) has to be made for

implementing it. For uniformity in the proof, the same A-construction

is made for SNS, QPL-2 and QSN relations also.

Thus, we are left only with the implementation of unary relations in sequence containing the I-type and E-type operators in general. The general graph-theoretical structure of these unary relations occurring in sequence, is a bouquet (see Section 4(f)(iv)

of Part IV). In such a case, some special considerations have to be taken into account. Each unary path contained in a unary bouquet (from input to a final output of the bouquet), as shown in Fig.8 of Part IV, may have any number of relations of the four types SNS, QPL-2, QSN and QPL-1, in any sequence. Since the algebra of SNS, QPL-2 and QSN are identical (as seen above) except for the nature of the matrix (i.e. 2X2, 3X3, 2X3 or 3X2), we shall club these together, and call them as Type-2, and call QPL-1 statements as Type-1, for the discussion below in the next subsection (c).

(c) Implementation of a unary path

Suppose we have a sequence of unary relations in a unary of the nature given below in (67 a to g),

$$\underline{\underline{a}} \quad \underline{\underline{z_1}} = \underline{\underline{b}} \quad (67a)$$

$$\underline{\underline{b}} \quad \underline{\underline{z_2}} = \underline{\underline{c}} \quad (67b)$$

$$\underline{\underline{c}} \quad \underline{\underline{z_3}} = \underline{\underline{d}} \quad (67c)$$

$$\mathcal{V}1(\underline{\underline{d}} \quad \underline{\underline{z_4}} \quad \underline{\underline{e}} = \underline{\underline{f}}) \quad (67d)$$

$$\mathcal{V}2 (\underline{\underline{f}} \underset{\sim}{\approx} \underline{\underline{g}} = \underline{\underline{h}}) \quad (67e)$$

$$\underline{\underline{h}} \underset{\sim}{\approx} \underline{\underline{k}} = \underline{\underline{l}} \quad (67f)$$

$$\underline{\underline{k}} \underset{\sim}{\approx} \underline{\underline{l}} = \underline{\underline{m}} \quad (67g)$$

We consider first a practical example like this to illustrate the problems involved and to discuss them. We shall give a general procedure for all such cases, after discussing the particular example.

In the sequence, putting in the SNS input $\underline{\underline{a'}}$ in the SNS-QPL relation $\underline{\underline{Z1}}$, the QPL output $\underline{\underline{b'}}$ can be worked out from (67a); which, on being put in the QPL-2 relation (67b), gives the QPL output $\underline{\underline{c'}}$. This again, on being input into the QPL-SNS relation (67c), gives the SNS output $\underline{\underline{d'}}$.

In the next QPL-1 equation, $\underline{\underline{d'}}$ which is an SNS term can be put as an input and the output $\mathcal{V}1'(\underline{\underline{f'}}) \equiv \underline{\underline{f'}}$ can be obtained from (67d). (This can be done, either by inputting $\underline{\underline{d'}}$ inside the

bracket on the l.h.s. quantified by $\mathcal{V}1$, or as a QPL-2 input

$\underline{\underline{d}}'$, by converting $\underline{\underline{d}}'$ to $\underline{\underline{d}}$ ' as in Eq.(29a to d), and then

converting $\underline{\underline{d}}'$ of QPL-2 to $\underline{\underline{d}}$ of QPL-1 using the standardizer.

Either procedure gives the same result, as can be checked out

in detail.) However, as discussed in Part IV, there is no

guarantee that this will go through the next QPL-1 equation in

(67e). For two unary QPL-1 relations in succession to be

implemented in general, they have to be first combined theoretically

to form a single QPL-1 relation, which is then implemented by

putting an input and taking out the output (as shown in Part IV).

Hence the correct procedure is not to input $\underline{\underline{d}}'$ in (67d), but to

combine (67d) and (67e) to get one equation - namely

$$\mathcal{V}3 (\underline{\underline{d}} \underset{\sim}{\sim} \underline{\underline{g}} = \underline{\underline{h}}) \quad (68a)$$

In this, if we input $\underline{\underline{d}}'$, or the corresponding $\underline{\underline{d}}'$, then we get

the output

$$\mathcal{V}3'(\underline{\underline{h}}') \equiv \underline{\underline{h}}' \quad (68b)$$

Thereafter $\underline{\underline{h'}}$ can be input into the QSN relation (67f) to obtain $\underline{\underline{k'}}$, and then again be input to the SNS relation (57g) to get finally $\underline{\underline{\ell'}}$.

The above description may look very complicated, but the essential principle to be grasped is very simple — namely that the Type-1 unary relations cannot be implemented one after the other, and may lead to the indefinite state, even though the combined statement corresponding to two statements in succession may lead to a definite state by theory. Therefore, the Type-1 statements should be combined theoretically if they occur in succession (2, 3 or any number of relations in continuous succession) and the equivalent single QPL-1 relation thereby obtained should be put in place of the number of successive QPL-1 relations.

On the other hand, QPL-2, SNS, QSN relations can be implemented one after the other by putting in the input and finding out the

output for one relation after the other. Therefore, the right procedure is first to examine the unary statements in the unary path, and check whether there occur two or more successive QPL-1 statements at any place. If so, they are first theoretically combined to make them into a single QPL-1 statement. We then replace the two or more successive statements of Type-1 by a single statement of Type-1. Thereafter, all the statements in the path can be implemented one after the other, by putting in the input state of each statement (starting from the first) and putting the output of this as input to the next successive statement, and carrying out this procedure in a serial order upto the end of the particular unary path.

Therefore, in summary, the only caution that is necessary in the implementation of a unary path is to note that the obvious method, of putting in the input in a relation and taking the output and fitting it to the next relation, is valid only for

Type-2 unary statements, and not for Type-1 unary statements.

Therefore, Type-1 statements in succession must be combined first

using the theoretical procedure given in Part IV, Section 4(f)(i)

(which may involve inverting them as in (65)), and the equivalent

single QPL-1 statement obtained thereby substituted in the

sequence, before the path is implemented in a practical example,

by putting in the actual state of the input of the first statement.

(d) Review of subsections (a), (b) and (c) and implementation of unary bouquets.

Thus, we have obtained the procedure for implementing a

binary forward statement (which can always be done by putting in

the inputs and taking out the output) for all four types, SNS,

QPL-2, QSN and QPL-1. We have also seen that binary reverse

statements can be converted to unary statements, in general.

While doing this conversion, if the term $\underline{\underline{c}}$ in $\underline{\underline{c}} \overset{\leftarrow}{Z} \underline{a} = \underline{b}$ is not

an original input, then a procedure analogous to the $\underline{\underline{c}}$ -reconstruction

in Fig.6 of Part IV will have to be made. On doing this, we obtain

a unary statement from the binary reverse statement, which may be of the E-type, A-type or I-type. Of these, the A-type is fed in as a reconstructed binary forward A-relation and hence only the E-type and I-type unary relations occur along unary paths in the form of bouquets.

For an indication of the nature of the problem that arises in implementing a unary bouquet, we shall take the example given in Fig.8(a) and (b) of Part IV. In this, all the unary paths originate from the same term t_1 and terminate in t_4 , t_6 , t_7 , t_8 and t_9 . As mentioned in Part IV, the path from t_1 to each terminus t_j is a unique one. (This is best noted by tracing the unary path in the inverse direction from the terminus to the origin.) We shall therefore show how each of these paths can be implemented and the output t'_j obtained, when we are given the input t'_1 for the origin term t_1 .

It must be mentioned that the procedure that we are indicating

is only one of several ways in which the process of implementing the bouquets can be done. Our attempt is to give a theoretical indication that the implementation can be carried out; but not necessarily to indicate the most effective, or the least time-consuming, process for a practical algorithm or computer program. In other words, we only try to give an existence proof that the unary bouquets can be worked out; but our method of proof is also practical.

Along each of the paths from t_1 to t_j in a bouquet, there can be unary relations of all different types, as in Eqs. (67a to g). As mentioned in that connection, the Type-2 relations can all be processed one after the other in a practical way, so that it is only necessary to combine the Type-1 (QPL-1) relations in order to make the path completely implementable. Hence the path is divided into a sequence of Type-1 and Type-2 statements — say as below:

$$\underline{t}_1 - (\text{Type-2 sequence}) - (\text{Type-1 sequence}) - (\text{Type-2 sequence}) \dots$$
$$(\text{Type-1 sequence}) - \underline{t}_j \quad (69)$$

In these, each Type-1 sequence is examined and if it has more than one Type-1 elementary relation, then all the relations in the sequence are combined theoretically by the method indicated in Section 4(f) of Part IV and the single resultant Type-1 relation is written. This is done for each one of the Type-1 sequences. Thereafter, we will have a sequence from t_1 to t_j in (69) in which every elementary relation is implementable from input t_1 to the output term t_j in continuous sequence. Hence, given the input t'_1 to the origin of the path, the final output t'_j along that particular path can be obtained. This can be repeated for each one of the terminal terms t_j (e.g. t_4 , t_6 , t_7 , t_8 and t_9 of Fig.8 of Part IV) of the bouquet. Hence the whole bouquet can be implemented from the given input state t'_1 of the term t_1 .

4. Flow chart for the implementation of a general argument containing all four types of elementary statements.

In this section, we shall show that the flow chart shown in

Fig.9 of Part IV for an argument containing QPL-1 statements

can be generalized for an argument containing all four types of statements — namely SNS, QPL-1, QPL-2 and QSN. In the last Section 3, we have shown that binary forward statements are implementable for all four types. We have also shown that unary bouquets are implementable, under all conditions. Therefore, it is only necessary to show that an argument in general can be converted into a form in which it contains only these two types of statements and that circular paths can be eliminated, multiple inputs to a term can be converted to a form with only two inputs at a time, contradictions checked for, and so on. We give below, in Fig. 1, the diagram of the flow chart for this in a form similar to Fig.6 of Part IV. There are only minor differences between the earlier flow chart for QPL-1 statements, and the present one. However, in order to make the chart quite general, terms and connective operators are given the general symbols t, a, etc, and Z, A, etc in the steps A to I of the chart.

The application to each of the four types of statements is indicated in the text that follows the flow chart under the headings A to I.

Fig.1 Flow Chart for the Implementation of a General Argument in Logic (for one variable).

(a) Notes and comments on the flow chart

A. In the flow chart we give only a procedure for working out the outputs $t_j^{(o)}$ and the additional outputs $t_j^{(ao)}$ for a given logical graph corresponding to given $t_j^{(i)}$ and $t_j^{(ai)}$ as in

H below. We do not consider the process of converting the logic of an argument given in words into the symbolic form we need.

This is not quite straightforward and requires some technical considerations. These are, however, omitted here and will be

(See, however, Section 4 for examples).
considered in a future report/ As mentioned above, this flow

chart only gives the procedure of working out a good logical graph from the inputs that are provided. As indicated in the introductory Section 1 and in the title of Fig.1, the QPL-1

Fig.1 Flow Chart for the Implementation of
a General Argument in Logic (for one variable)

($\underline{t}^{(i)}$) = input term, ($\underline{t}^{(m)}$) = middle term, ($\underline{t}^{(o)}$) = output term;
 $t^{(ai)}$, $t^{(am)}$, $t^{(ao)}$ = "additional" input, middle, output
 terms; Z = Connective operator; V = vidya operator in
 general (may be \underline{V} or \overline{V}) for consistency check.)

A. Write down the elementary steps of the argument as provided. Draw the corresponding logical graph, and from its structure, list the terms as $\underline{t}_j^{(i)}$, $\underline{t}_j^{(m)}$ and $\underline{t}_j^{(o)}$, and list all Z_j . Call this List A.



B. Check for the occurrence of directed circuits. For each one present, make the "circuit removal construction" as in Fig.4(b) of Part IV for each circuit. Add the vertices corresponding to the new terms $\underline{t}^{(i)}$, $\underline{t}^{(m)}$, $\underline{t}^{(o)}$ and the new operators Z_j (of the type V) to List A. Call the new list as List B.



C. Make the V -construction for multiple inputs to a term \underline{t}_j , as in Eqs (49) and (50) of Part IV, and do this for all such terms. Add the additional Z_j (of the type V) and the $\underline{t}_j^{(m)}$ and $\underline{t}_j^{(o)}$ generated to List B. Call the new list as List C.



Fig.1 Flow Chart (Contd.)

D. List the binary reverse relations. If any of these does not have \underline{c} as a $t^{(i)}$, carry out the \underline{c} -reconstruction (see text). Add, or replace the operators Z_j so produced and add the additional terms $t^{(ai)}$, $t^{(ao)}$ and also replace the relevant $t^{(m)}$ in the list C. Call the new list as List D.

E. Input the states given for $t^{(i)}$. For $t^{(ai)}$ choose one set of possible inputs, using one of the basic states $-T, F$ or \forall, Σ, Φ . Call this as Set 1, and for the later trials Set 2, Set 3, and so on, for List D.

F. Make the modified A-construction, as in Fig.6(b) of Part IV, for all unary relations of the A-type. Add the $t^{(am)}$ and $t^{(ao)}$ so produced to list D. Add also the operators Z_j (of type V) that are introduced, to List D. Call the new list as List F.

G. List out the unary paths from origin to each terminus in the unary bouquets and reduce the QPL-1 sequences to single QPL-1 relations. Eliminate the $t_j^{(m)}$ and Z_j so removed and add the new Z_j so introduced to List F. Call this as List G. (Obtain thus the List of terms and operators of the "reduced graph", containing only binary forward relations and unary paths that are implementable sequentially.)

H. Implement the List G of the reduced graph, rearranging the sequence in the list if needed, and obtain the logical states of all outputs $t^{(o)}$ and $t^{(ao)}$. This is for Set 1 of the inputs $t^{(i)}$ and $t^{(ai)}$ as in E, for List D.

I. Repeat the steps E, F, G, H, for Set 2, Set 3, ..., of $t^{(ai)}$ and list the $t^{(o)}$ and $t^{(ao)}$ for each set, as desired.

statements are all supposed to be formulated for one variable x.

As indicated in Eq (35), a relation in QPL-2 of the type,

$$(\nexists/x) (\underline{a}(x)) \& \nexists/(x) (\underline{b}(x)) \Rightarrow (\nexists/x) (\underline{c}(x)) \quad (69a)$$

has, in so far as its logic is concerned, the same format as for

$$(\nexists/x) (\underline{a}(x)) \& \nexists/(y) (\underline{b}(y)) \Rightarrow (\nexists/z) (\underline{c}(z)) \quad (69b)$$

So also,

$$(\exists x) (\underline{a}(x)) \Rightarrow (\exists x) (\underline{b}(x)) ; (\exists x) (\underline{a}(x)) \Rightarrow (\exists y) (\underline{b}(y)) \quad (70a,b)$$

have also the same formulae as far as the matrix logic is concerned.

Thus, many of the equations in the argument could also be implemented for functions (terms) having the same, or different variables, but with each being only a function of one variable.

Thus, in this report, we only assure that all the theory that is formulated is valid for terms which are functions of a single variable for QPL statements, in association with any number of SNC statements having no variables attached to them. The theory of this report may be considered to be restricted to logical terms which are functions of one variable.

b. As mentioned in connection with the flow chart in Part IV, we should mention that the exact position where the break in a directed circuit is made, has to be specified beforehand. We have not proved that the properties of a circuit will remain the same when the check for consistency is made at one term instead of another term in the path. This again is a matter that has to be studied further (see, however, Section 4(b)).

c. In removing multiple inputs to a term \underline{t}_j and replacing them by a series of binary inputs and checking them for consistency, we follow the same procedure as in Section 4(c) of Part IV, where this was described for QPL terms, using the QPL vidya operator \bigvee . It is obvious that, if the term \underline{t}_j is an SNS term, the same equations will hold good except that the consistency check operator will be the SNS vidya operator, namely \bigvee_{S} .

D. The \underline{c} -reconstruction process is quite similar for all types of relations and we shall write them for three particular equations that require this process, to illustrate its variations.

The reconstruction in the logical graph is similar in all cases.

SNS:

$$\underline{c} \xleftarrow{\quad} \underline{a} = \underline{b} \rightarrow \underline{a} \approx \underline{b} \text{ for } \underline{c}' = T; \rightarrow \underline{a} \gtrapprox \underline{b} \text{ for } \underline{c}' = F \quad (71a)$$

QPL-1:

$$\begin{aligned} \mathcal{V}(\underline{c} \xleftarrow{\quad} \underline{a} = \underline{b}) &\rightarrow \mathcal{V}'(\underline{a} \approx \underline{b}) \text{ for } C' = \mathcal{V}_c(T); \mathcal{V}' = \mathcal{V}_c \oplus \mathcal{V} \\ &\rightarrow \mathcal{V}(\underline{a} \gtrapprox \underline{b}) \text{ for } C' = \mathcal{V}_c(F); \mathcal{V}' = \mathcal{V}_c \oplus \mathcal{V} \end{aligned} \quad (71b)$$

QPL-2:

$$\underline{c} \xleftarrow{\quad} \underline{a} = \underline{b} \rightarrow \underline{a} \approx \underline{b} \text{ for } \underline{c}' = T; \rightarrow \underline{a} \gtrapprox \underline{b} \text{ for } \underline{c}'' = F \quad (71c)$$

For SNS-QPL and QPL-SNS, the equations are closely similar to (71a) and (71c), and are obvious.

E. Note that the input states $t^{(ai)}$ need only be one or the other of the basic states, namely $T(1 \ 0)$ and $F(0 \ 1)$ for SNS, and $\mathcal{V}(1 \ 0 \ 0)$, $\mathcal{S}(0 \ 1 \ 0)$, $\mathcal{P}(0 \ 0 \ 1)$ for QPL-2.

Similarly, for QPL-1, only the four QPL states namely $\forall(T)$,

$\forall(F)$, $\exists(T)$ and $\exists(F)$ need be tried. We will finally obtain

the list of outputs and additional outputs corresponding to all

of these. Only those for which there are no contradictions in the

outputs, or additional outputs, need be considered; and, from a list

of these $t^{(z_0)}$ and $t^{(o)}$, it would be possible to work out which

sets of outputs are permissible for a given argument. It has to

be noted that for a given set of $t^{(i)}$, there may be more than

one set of $t^{(o)}$ that are permissible depending on the choice of

$t^{(ai)}$, which becomes necessary because of the procedures described

in B, C and D. (See below under H).

F. No special notes need be made except that the added operators of type \underline{V} may be either $\underline{\underline{V}}$ or $\underline{\vee}$, depending upon the nature of the term which is involved in the A-construction.

G. The reduced graph is always implementable by the method of proof that was given in Part IV. However, in practice, if

necessary, the sequence of the listing of elementary relations may have to be rearranged so that the implementation can go on in continuous sequence from one equation to the next. The nature of a practical algorithm for this purpose is indicated in Section 4(c) by taking an example and the principles involved in generalizing it is indicated there.

H and I. The rearrangement of the list has been commented upon under G. The way of interpreting the outputs and additional outputs has also been commented upon under E. The only new comment that is needed is the following: We first implement List G for Set 1 chosen under the step E, and the outputs are printed, along with the additional outputs. The procedure from E to H is repeated for Set 2, Set 3 and so on and the list of $\underline{t}^{(ai)}$'s that were chosen, and the corresponding $\underline{t}^{(o)}$'s and $\underline{t}^{(ao)}$'s, are printed in a tabular form. We shall indicate in Section 4(c) how such a table can be interpreted for the nature of the outputs of an argument, for the given set of inputs.

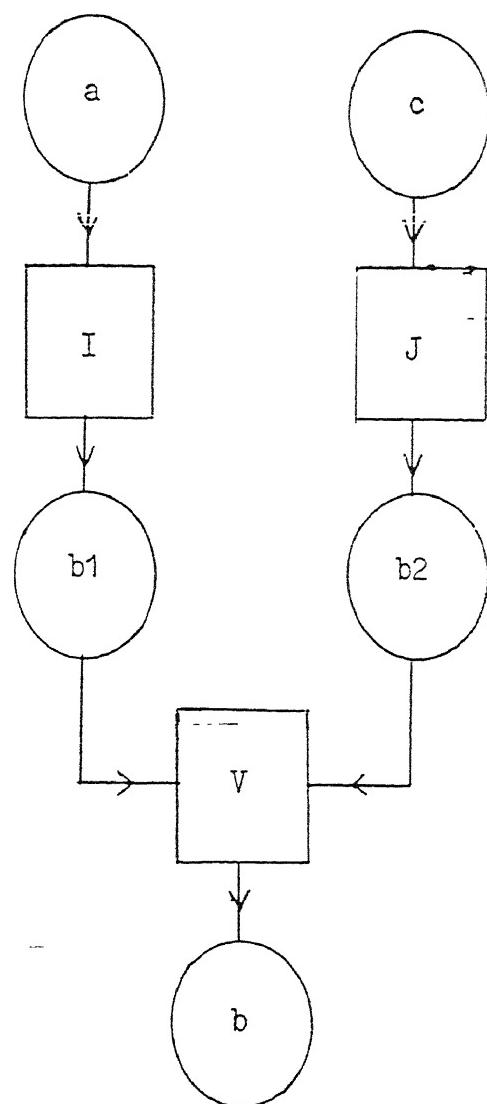
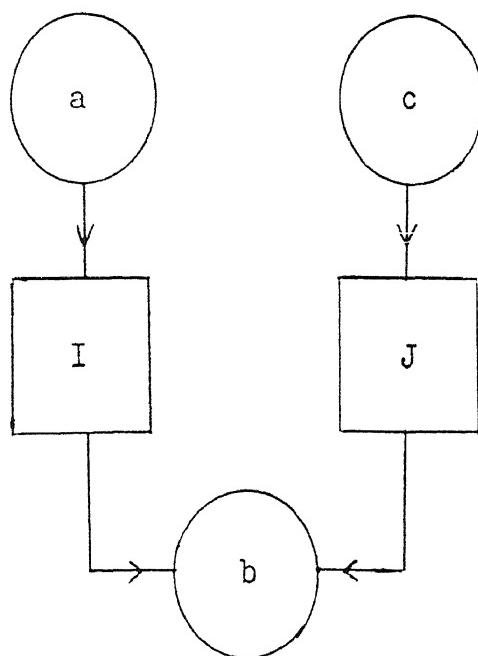
(b) Setting up of the flow chart of an argument

As mentioned in the introduction and in the comments relating to A of the flow chart, the flow chart only assures that given a set of elementary statements forming an argument or the logical graph of the argument, we can always implement these statements so as to obtain the states of the outputs in the flow chart, given the input states into the argument. However, there are some problems involved in writing down the set of elementary statements that describe the logic of a given argument in the most efficient manner. We shall illustrate this by a simple example of an argument consisting of two unary statements, namely

$$\underline{a} \Rightarrow \underline{b}, \quad \neg \underline{c} \Rightarrow \neg \underline{b} \quad (72a, b)$$

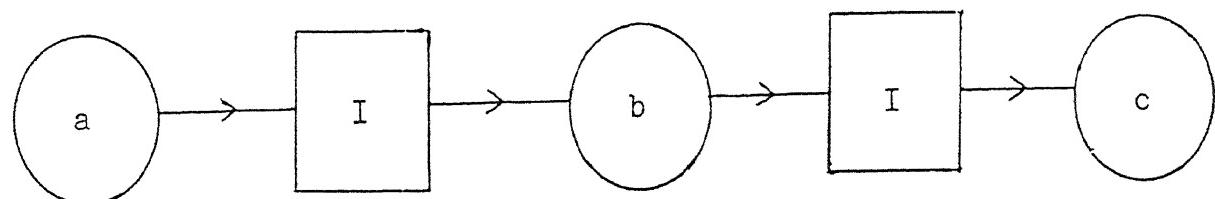
The logical graph of this is shown in Fig. 2(a).

- Fig.2 Logical graph of a simple argument consisting of two steps
- (a) As given in Eq (72 a & b).
 - (b) The same after $\underline{\pi}$ -construction to remove multiple inputs.
 - (c) The same, after inverting the second Eq (72b) to obtain the pair of Eqs (73a, b).



(a) As given in Eq (72 a, b)

(b) The same after V-construction to remove multiple inputs



(c) The same, after inverting the second Eq (72b) to obtain the pair of Eqs (73a, b).

Since there are two inputs to the term \underline{b} , under the instructions in Step C of the flow chart, a $\underline{\underline{V}}$ -construction will have to be made and we obtain the graph in Fig. 1(b).

Suppose that we are given the input $\underline{a} = T$, and we are required to find the output state of \underline{c} . We can do this by inputting the states T and F for \underline{c} in Fig. 2(b) and finding out which state⁽¹⁾ lead to a non-contradictory state for \underline{b} . This is shown schematically in Table 1.

Table 1. Checking for non-contradictory inputs for \underline{c} in Fig.2(b).

On examining Table 1, it will be seen that $\underline{c} = F$ leads to the contradictory state X for \underline{b} and is hence disallowed. Consequently, if only the states T and F are allowed for \underline{c} (as mentioned in Step E of Flow Chart for the states to be tried) it follows that \underline{c} can only have the state T. Thus, the logical graph of Fig.2 leads to the result that \underline{c} can only be T..

Table 1. Checking for non-contradictory inputs for $\underline{\underline{c}}$
in Fig. 2(b)

(a) For Equations (72a,b), with $\underline{\underline{a}} = T \rightarrow \underline{\underline{b1}} = T$			
$\underline{\underline{c}}$	T	F	D
$\underline{\underline{b2}}$	D	F	D
$\underline{\underline{b}}^*$	T	X	T
Conclusion	$\underline{\underline{c}} = F$ leads to $\underline{\underline{b}} = X$ and is forbidden, so that $\underline{\underline{c}} = T$.		

(b) For Equations (74a,b) with $\underline{\underline{a}} = T \rightarrow \underline{\underline{b1}} = F$			
$\underline{\underline{c}}$	T	F	D
$\underline{\underline{b2}}$	D	F	D
$\underline{\underline{b}}^*$	F	F	F
Conclusion	Both $\underline{\underline{c}} = T$ and $\underline{\underline{c}} = F$ are possible, so that $\underline{\underline{c}} = D$.		

$$* \quad \underline{\underline{b1}} \vee \underline{\underline{b2}} = \underline{\underline{b}}$$

However, in SNS logic, we can also have the input $\underline{c} = D$ in Fig. 2(b), when we get the result shown in column 3 of Table 1.

We find that there is no contradiction and therefore the state D for \underline{c} is also permissible. However, this is reasonable, since $D = T \oplus F$, and since (72b) is permissible for $\underline{c} = T$, it will also give a non-contradictory state for \underline{b} , for $\underline{c} = T$ or $F = D$, and nothing new is obtained from column 3.

Now, we shall re-write Eq. (72a,b) by inverting the second equation, when we obtain the two logical equations

$$\underline{a} \Rightarrow \underline{b}, \quad \underline{b} \Rightarrow \underline{c} \quad (73a,b)$$

corresponding to the logical graph shown in Fig. 2(c). If the input $\underline{a} = T$ is put in, then by the first implication \underline{I} , we obtain $\underline{b} = T$, and from it, by the second implication \underline{I} , we obtain $\underline{c} = T$. Thus, we find that, when the second equation (72b) is inverted and worked out, then the state of the output \underline{c} so obtained (necessarily T) is the same as what was obtained from (72a,b),

via the V-construction for consistency check. In other words,

the result obtained is the same, whether the second equation in (72) and (73) is taken in the direct or inverted forms.

We shall now make a similar check for the pair of equations in (74a,b) below, with the second equation being inverted in (75b), but with the input to it being $\neg \underline{b}$ (or $\underline{b} = F$).

$$\underline{a} \Rightarrow \neg \underline{b}, \quad \neg \underline{c} \Rightarrow \neg \underline{b} \quad (74a,b)$$

$$\underline{a} \Rightarrow \neg \underline{b}, \quad \underline{b} \Rightarrow \underline{c} \quad (75a,b)$$

By making a V-construction as in Fig. 2(b), we get the results

shown in Table 1(b), for the possible inputs of \underline{c} . In this case, with $\underline{b}_1 = F$, both the pure states T and F of \underline{c} do not lead to a contradiction and are therefore permissible for \underline{c} . Hence, the output for the pair of equations (74a,b) is:

$$\underline{c} = T \underset{\approx}{\cup} F \equiv T \oplus F = D \quad (74)$$

Obviously, as shown in the third row of Table 1(b), $\underline{c} = D$ is itself consistent for this pair of equations.

Now, if we consider the pair (75a,b), we again obtain

$\underline{c} = D$ as follows (in agreement with (74)):

$$\underline{a} = T, \underline{a} \underset{\sim}{\mathcal{I}} \underline{N} = \underline{b} \rightarrow \underline{b} = F; \quad \underline{b} = F, \underline{b} \underset{\sim}{\mathcal{I}} \underline{c} \rightarrow \underline{c} = D \quad (75a,b)$$

Thus, the unary relation of SNS implication

$$\underline{b}', \quad \underline{b} \underset{\sim}{\mathcal{I}} \underline{c} \rightarrow \underline{c}' \quad (76)$$

leads to the same result for the state of \underline{c}' , for both $\underline{b}' = T(1 \ 0)$

and $F(0 \ 1)$, irrespective of the sense in which the relation is

worked out (from \underline{b} to \underline{c} in (76), or equivalently as $\underline{c} I^t = \underline{b}$,

from \underline{c} to \underline{b}). We have proved this above in detail, for $\underline{b}' = T$

or $\underline{b}' = F$ and hence it is true also for $\underline{b}' = D$.

In the same way, any unary $\underset{\sim}{\mathcal{I}}$ -type relation in QPL-2, in the forward direction, namely

$$\underline{b}', \quad \underline{b} \underset{\sim}{\mathcal{I}} \underline{c} = \underline{c} \rightarrow \underline{c}' \quad (77)$$

will give the same result for \underline{c}' as the Eqs (78a,b,c) below, with

the relation written in the inverted form from \underline{c} to \underline{b} :

$$\underline{b}' = \underline{b}_1, \quad \underline{c}' \underset{\sim}{\mathcal{I}}^t = \underline{b}_2, \quad \underline{b}_1 \vee \underline{b}_2 \neq \emptyset \quad (78a,b,c)$$

We shall give an example and indicate the nature of a general proof of this in Section 5(a), as well as its extension to QSN relations.

The extension to the E-type relations of the identity of the output, when $\underline{b} \underline{E} = \underline{c}$ (or $\underline{b} \underline{N} = \underline{c}$) is applied in the forward, or reverse, sense, is quite straightforward. Also, as indicated in Section 5 , the extension to $\underline{\underline{E}}(i, j)$ and $\underline{\underline{E}}^C(i, j)$ in QPL and similar operators in QSN is readily proved. The A-type relations are always worked out only using the A-construction in unary relations (see Step F of flow chart), so that the above proof is unnecessary in this case.

This study thus gives a confirmation of the general principle proved in Part I [1] , namely that the inverse of a relation \underline{Z} , with matrix $|Z|$, has the matrix $|Z^t|$, since the above treatment in SNS for 2X2 matrices is readily generalizable to $m \times n$ matrices.

Therefore, if we have an argument in which a final output

(for example \underline{x}) is not the output of a statement such as $\underline{g} \underline{I} = \underline{x}$

or $\underline{g} \underline{A} \underline{b} = \underline{x}$, but is of the form $\underline{x} \underline{J} = \underline{y}$ or $\underline{x} \underline{A} \underline{b} = \underline{g}$, then

\underline{x} is taken as an additional input ($\underline{t}_j^{(ai)}$), and the permissible

states of \underline{x} are worked out, as indicated in the flow chart.

Putting in $\underline{x}' = T(1 \ 0)$ and $F(0 \ 1)$ for SNS terms, and $\underline{\underline{x}}' =$

$\nabla(1 \ 0 \ 0)$, $\zeta(0 \ 1 \ 0)$ and $\phi(0 \ 0 \ 1)$ for QPL terms, the

permissible pure states of \underline{x} can be worked out as per the procedure

indicated in the flow chart. Then, the output state for \underline{x}' is the

Boolean sum of the permissible pure states thus obtained, using

the logical operator $\underline{\cup}$ or $\underline{\mid\cup}$ as the case may be. (Note: The

adjective "pure" is used for the basic states of BA-2 or BA-3,

to distinguish them from "mixed" states, which are Boolean sums

of two or more pure states. This follows the terminology for

vectors representing pure and mixed states in quantum mechanics.)

The above considerations also give a justification of the procedure stated in Step 4 of the Flow Chart for all additional inputs $\underline{t}^{(ai)}$.

(c) Rearrangement of the logical equations of a reduced graph
to be sequentially implementable

We shall ^{now} illustrate the principles of writing down the logical equations of an argument in sequential order, so as to be implementable straightforwardly one after the other. For this, we take an argument in SIS, as the principles of graph construction depend only on the unary or binary nature of each elementary equation, and does not depend on whether it is in SNS, QPL or QSN. We shall show how the reduced graph in Step F of the Flow Chart is converted into such a set of implementable equations. We have also chosen an argument without any binary reverse equations, as this requires the examination of two possibilities, namely $\underline{c} = T$, and F , in $\underline{c} \xrightarrow{\underline{Z}} \underline{a} = \underline{b}$. So also, the A-construction is avoided. (Later, we shall indicate how both these specialities can be built in and worked out in an extension of this argument.) The set of equations of this small argument, which has, however, multiple inputs to a term, is given in (79). We then indicate

how two V-constructions added to it, as in (80), make it into a reduced graph. Thereafter, we shall describe the algorithmic process of writing the logical equations for this reduced graph, so as to be sequentially implementable.

(i) Simple example of an argument without circularity

The argument has the logical equations as in (79a to f):

$$\underline{a} \underset{\sim}{\wedge} \underline{b} = \underline{g} \quad (79a) ; \quad \underline{z} \underset{\sim}{\wedge} \underline{N} \underline{I} = \underline{g} \quad (79b)$$

$$\underline{c} \underset{\sim}{\wedge} \underline{I} = \underline{h} \quad (79c) ; \quad \underline{x} \underset{\sim}{\wedge} \underline{J} = \underline{g} \quad (79d)$$

$$\underline{g} \underset{\sim}{\wedge} \underline{h} = \underline{k} \quad (79e) ; \quad \underline{g} \underset{\sim}{\wedge} \underline{N} \underline{K} = \underline{y} \quad (79f)$$

We are further given that \underline{a} , \underline{b} , \underline{c} are the inputs, and have the states T, F, T respectively, and we wish to find the states of the outputs \underline{x} , \underline{y} , \underline{z} . Since there are multiple inputs to \underline{g} , we rewrite the equations (79a), (79b) and (79d) with appropriate vidya operators, as below:

$$\underline{a} \underset{\sim}{\wedge} \underline{b} = \underline{g1} \quad (80a) ; \quad \underline{z} \underset{\sim}{\wedge} \underline{N} \underline{I} = \underline{g2} \quad (80b)$$

$$\underline{g1} \underset{\sim}{\wedge} \underline{g2} = \underline{g2}' \quad (80c) ; \quad \underline{x} \underset{\sim}{\wedge} \underline{J} = \underline{g3} \quad (80d)$$

$$\underline{g3} \underset{\sim}{\wedge} \underline{g2}' = \underline{g} \quad (80e)$$

where (80a), (80b), (80d) replace (79a), (79b), (79d), and (80c) and (80e) are added to the set of equations describing the argument. We then obtain a "reduced graph" of the argument, with only unary and binary forward operators, and this is shown in Fig. 3. It is a directed graph, and is a tree.

Fig.3. Reduced graph of the argument composed of Eqs. (79), modified as in (80).

The algorithm for writing the equations in sequence is as follows:

We first examine the graph for inputs (i), additional inputs (ai), final outputs (o), and middle terms (m), by the following procedure: The input terms $\underline{t}^{(i)}$ and $\underline{t}^{(ai)}$ are those that have no edges leading to them, and output terms $\underline{t}^{(o)}$ are those that have no edges going out of them. The middle terms have edges going into them and also coming out of them.

If this check is made on the graph in Fig.3 we obtain the following list:

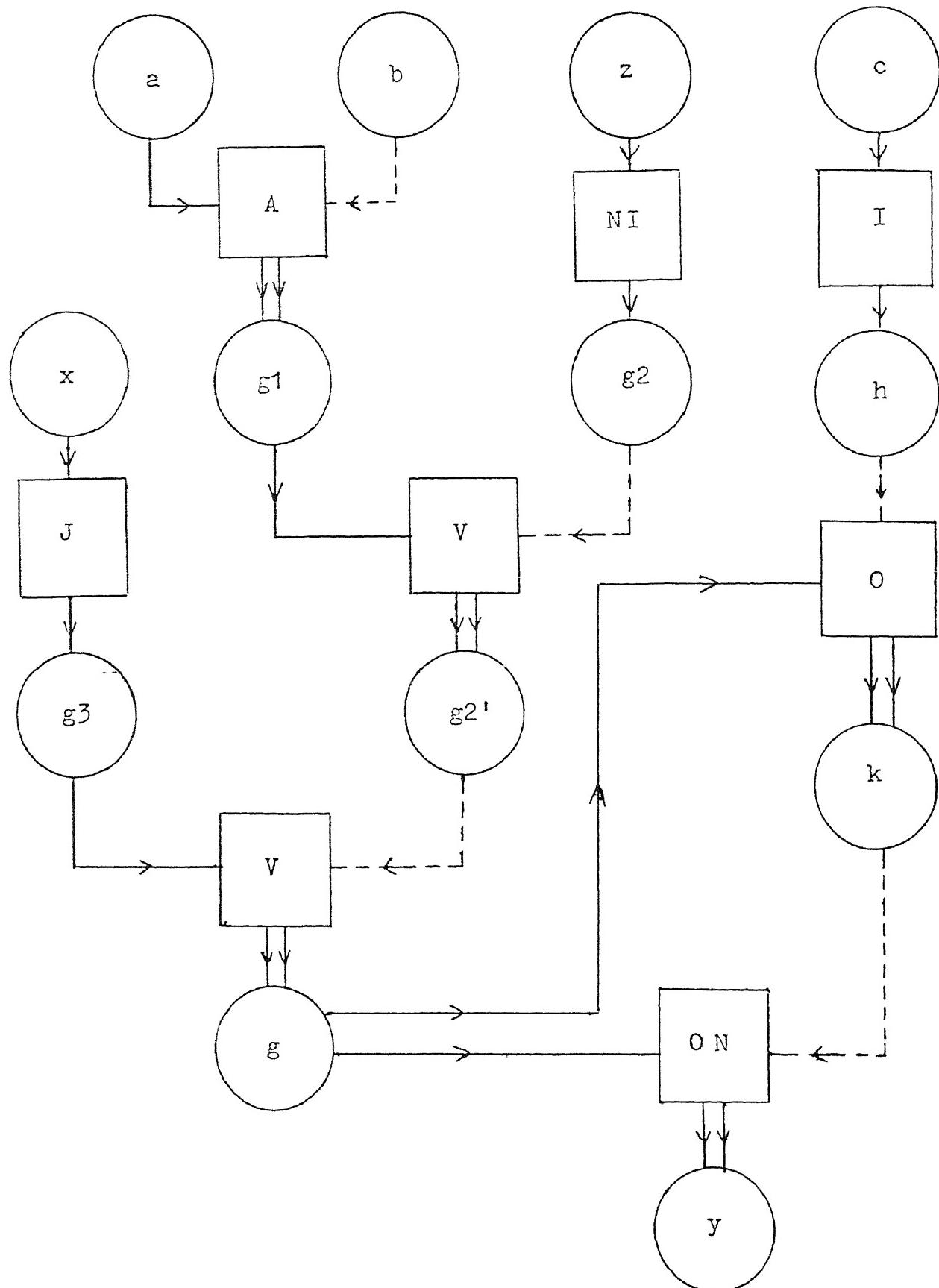


Fig.3 Reduced graph of the argument composed of Eqs. (79) modified as in (80).

Inputs ($\underline{t}^{(i)}$) : $\underline{\underline{a}}, \underline{\underline{b}}, \underline{\underline{c}}$

Additional inputs ($\underline{t}^{(ai)}$) : $\underline{x}, \underline{\underline{z}}$

Outputs ($\underline{t}^{(o)}$) : $\underline{\underline{y}}$

Middle terms ($\underline{t}^{(m)}$) : $\underline{\underline{g1}}, \underline{\underline{g2}}, \underline{\underline{g2}'}, \underline{\underline{g3}}, \underline{\underline{g}}, \underline{\underline{h}}, \underline{\underline{k}}$

We then proceed to write the logical equations in sequential order in a series of steps, numbered Step 1, Step 2, etc.

Step 1

Write down all the elementary equations with $\underline{t}^{(i)}$ and $\underline{t}^{(ai)}$ as inputs and write down the list of intermediate outputs of Step 1 (which may be called $\underline{t}^{(1)}$).

$$\underline{\underline{a}} \ A \ \underline{\underline{b}} = \underline{\underline{g1}} \quad (81a) ; \quad \underline{\underline{c}} \ I = \underline{\underline{h}} \quad (81b)$$

$$\underline{x} \ J = \underline{\underline{g3}} \quad (81c) ; \quad \underline{\underline{z}} \ NI = \underline{\underline{g2}} \quad (81d)$$

List of $\underline{t}^{(1)}$ for Step 1 : $\underline{\underline{g1}}, \underline{\underline{g2}}, \underline{\underline{g3}}, \underline{\underline{h}}$.

Step 2

Take each $\underline{t}^{(1)}$ and write down the elementary equations containing them as follows:

$$\underline{\underline{g}}^1 \underline{\underline{v}} \underline{\underline{g}}^2 = \underline{\underline{g}}^2' \quad (81e)$$

$$(\underline{\underline{g}}) \underline{\underline{Q}} \underline{\underline{h}} = - \quad (\text{Hold})$$

$$\underline{\underline{g}}^3 \underline{\underline{v}} (\underline{\underline{g}}^2') = - \quad (\text{Hold})$$

List of $\underline{\underline{t}}^{(2)}$ for Step 2 : $\underline{\underline{g}}^2'$

It will be noticed that the terms enclosed by brackets in the equations tested for Step 2 are not available either as original or additional inputs, or as intermediate outputs ($\underline{\underline{t}}^{(1)}$), of Step 1. Hence these equations are marked as "Hold" and are carried over to Step 3.

Step 3

Take the held over equations in the previous steps, as well as equations involving $\underline{\underline{t}}^{(2)}$ and write them as below.

$$\underline{\underline{g}}^3 \underline{\underline{v}} \underline{\underline{g}}^2' = \underline{\underline{g}} \quad (81f)$$

$$(\underline{\underline{g}}) \underline{\underline{Q}} \underline{\underline{h}} = - \quad (\text{Hold})$$

List of $\underline{\underline{t}}^{(3)}$ for Step 3 : $\underline{\underline{g}}$

In the same way as in Step 2, one equation has to be put in "Hold" in Step 3 also, since one of the terms in the equation is not contained in the list of inputs, or of $\underline{\underline{t}}^{(1)}$, $\underline{\underline{t}}^{(2)}$, of the previous steps.

Step 4

Similar to Step 3, we obtain

$$\underline{\underline{g}} \quad \underline{\underline{Q}} \quad \underline{\underline{h}} = \underline{\underline{k}} \quad (81g)$$

$$\underline{\underline{g}} \quad \underline{\underline{Q_N}} \quad \underline{\underline{(k)}} = - \quad (\text{Hold})$$

List of $\underline{\underline{t}}^{(4)}$ for Step 4 : $\underline{\underline{k}}$

Step 5

Doing the same as in Steps 3 and 4, we obtain

$$\underline{\underline{g}} \quad \underline{\underline{Q_N}} \quad \underline{\underline{k}} = \underline{\underline{y}} \quad (81h)$$

List of $\underline{\underline{t}}^{(5)}$ for Step 5 : $\underline{\underline{y}}$.

Thus, all the equations leading down to the final output

$\underline{\underline{y}}$ are obtained. The sequentially implementable equations are

given by (81a) to (81h) under the various steps discussed above.

It is verified that the number of equations (namely 8) is equal to the number of operators contained in the graph in Fig. 3.

Although the number of steps (namely 5) seems to be too large for this simple problem, it becomes reasonable for larger problems, and the principle of this algorithm is very general and the process can be used for any logical graph containing

unary and binary elementary statements (not necessarily a reduced graph but one without circular arguments). If an unary bouquet connecting \underline{a} say to $\underline{b}, \underline{c}, \dots, \underline{k}$ occurs, then all the unary paths in it originating from \underline{a} are taken to be implementable in a single "step" of the algorithmic process described above.

The essence of the algorithm is that, at each step (say Step n) only the input terms ($t^{(i)}$ and $t^{(ai)}$) and the output terms of the previous steps ($t^{(1)}, t^{(2)}, \dots, t^{(n-1)}$) are employed for testing the implementability of equations in that Step n , consisting of the held over equations of the previous step ($n-1$), and elementary equations containing the outputs $t^{(n-1)}$. Hence, every elementary statement in every step is implementable when the sequential processing of the statements reaches it.

(ii) Example with circularity, c-reconstruction and A-construction

The results of the implementation of the reduced graph in Fig.3 and its interpretation for the states of the outputs $\underline{x}, \underline{y}, \underline{z}$

are discussed in the next Section 4(d). Here we shall now consider an amplification of the graph of Fig. 3, in order to illustrate the Step B (circuit removal), Step D (c-reconstruction) and Step F (A-construction) of the Flow Chart. All these processes become necessary on changing one equation, namely (79b), as in (82) below.

$$\text{Replace (79b) by } \underline{y} \xleftarrow{\Omega^c} \underline{z} = \underline{g} \quad (82a)$$

$$\text{where } \underline{\Omega^c} \equiv \underline{\underline{\underline{\Omega}}}$$

This brings in a circuit composed of the following terms and operators:

$$-\underline{y} - \underline{\Omega^c} - \underline{g} - \underline{\underline{\Omega N}} - \underline{y} - \quad (83)$$

As stated in the explanation of Step B in Fig. 1, the exact point at which the circuit is broken has to be stated, and we choose it to be at the term \underline{y} , and write the relevant equations as

$$\underline{y_1} \xleftarrow{\Omega^c} \underline{z} = \underline{g} \quad (84a) ; \quad \underline{g} \xrightarrow{\Omega N} \underline{k} = \underline{y_2} \quad (84b)$$

$$\underline{y_1} \xrightarrow{\Omega^c} \underline{y_2} = \underline{y} \quad (84c)$$

We then remove the multiple inputs to $\underline{\underline{g}}$ (as per Step C of the Flow Chart) by introducing two vidya operators (exactly as in (80c) and (80e)), and then obtain the graph shown in Fig.4(a).

Fig.4(a) Graph of the argument composed of Eqs. (79), (80) and (82), after the "circuit removal construction" in Step E, and the "V-construction" for removing multiple inputs in Step C, of the Flow Chart.

- (b) c-reconstruction (Step D of Flow Ch rt) when $\underline{\underline{y}}_1 = F$.
(See also Fig. 7(b) of Part IV.)
 - (c) c-reconstruction (as in Step D of Flow Chart) when $\underline{\underline{y}}_1 = T$ and A-construction for NAN (as in Step F)
to $\underline{\underline{z}}_1 = F$ and $\underline{\underline{g}}_2 = F$. (See also Fig.6(b) of Part IV.)
-

It will be noticed that, in this graph, the circuit in (83) is

converted into a directed path from $\underline{\underline{y}}_1$ to $\underline{\underline{y}}_2$, namely

$\underline{\underline{y}}_1 \rightarrow \overbrace{\text{NAN}}^{\text{V}} \rightarrow \underline{\underline{g}}_2 \rightarrow \overbrace{\text{z}}^{\text{V}} \rightarrow \underline{\underline{g}}_2' \rightarrow \overbrace{\text{z}}^{\text{V}} \rightarrow \underline{\underline{g}} \rightarrow \overbrace{\text{ON}}^{\text{V}} \rightarrow \underline{\underline{y}}_2$, and a

consistency check $\underline{\underline{y}}_1 \stackrel{\text{V}}{\sim} \underline{\underline{y}}_2 = \underline{\underline{y}}$ between the origin $\underline{\underline{y}}_1$ and

the terminus $\underline{\underline{y}}_2$ of this path. Since the equations pertaining

to this graph are obtained by combining (79a-h), (80a-e, and

(84a-c), we shall rewrite them in (85) for ready reference:

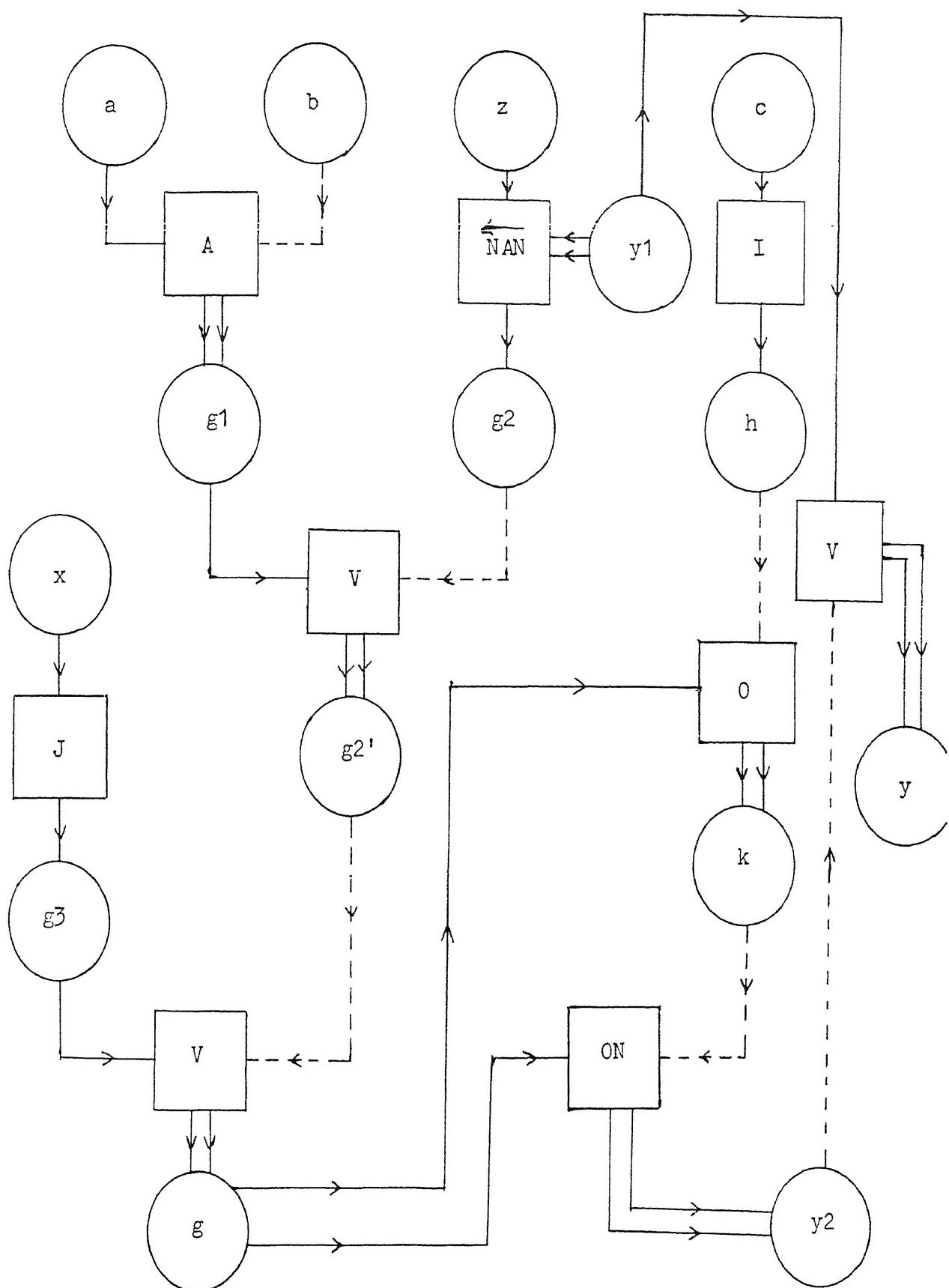


Fig.4(a) Graph of the argument composed of Eqs.(79),⁽⁸⁰⁾ and (82), after the "circuit removal construction" in Step B, and the "V-construction" for removing multiple inputs in Step C, of the Flow Chart.

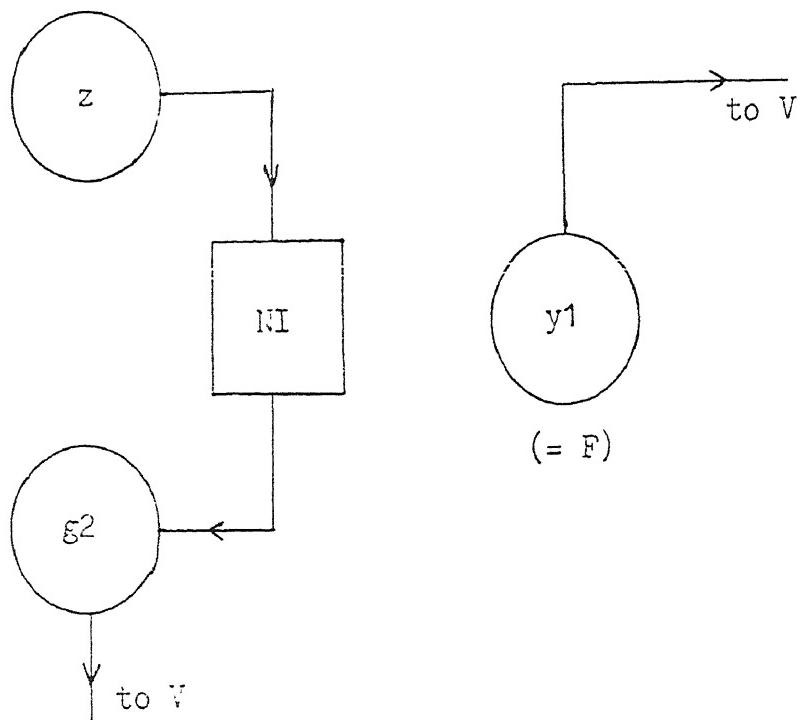


Fig.4(b). c-reconstruction (Step D of Flow Chart) when $\underline{\underline{y}}^1 = F$.
(See also Fig. 7(b) of Part IV.)

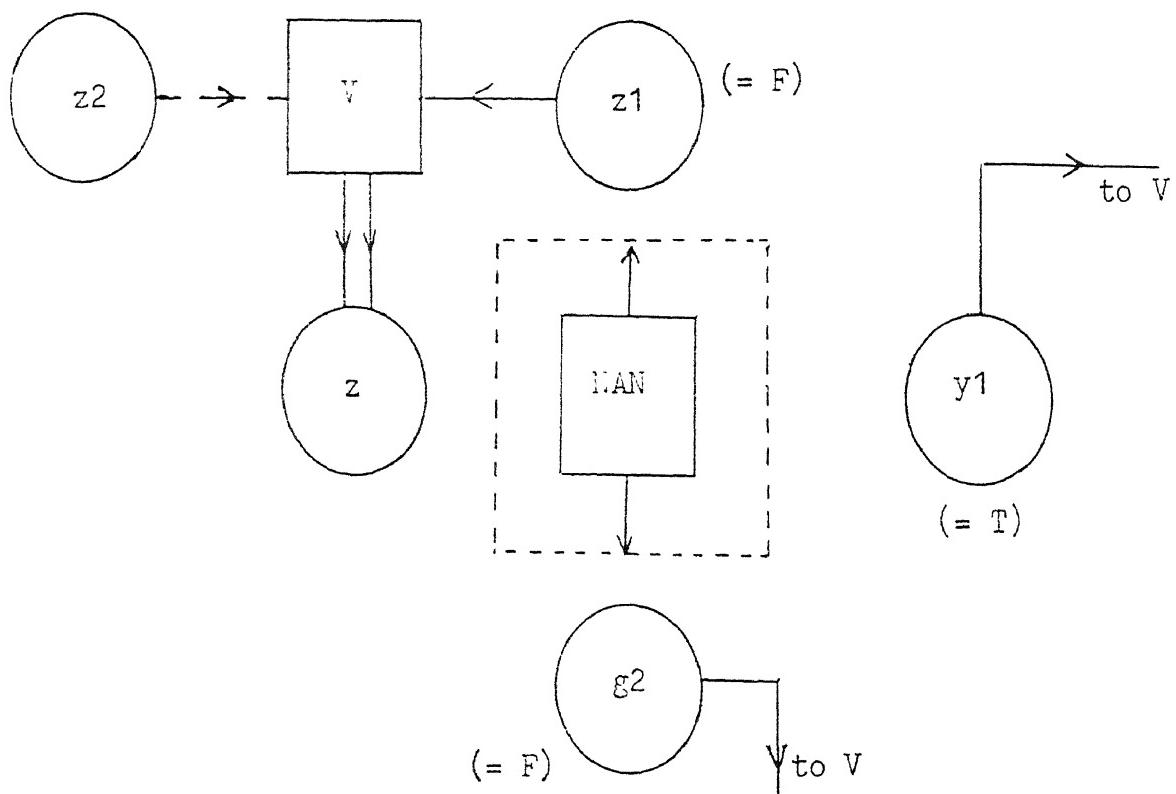


Fig.4(c). c-reconstruction (as in Step D of Flow Chart) when
 $\underline{\underline{y}}^1 = T$ and A-construction for NAN (as in Step F) leading
to $\underline{\underline{z}}^1 = F$ and $\underline{\underline{g}}^2 = F$. (See also Fig.6(b) of Part IV.)

$$\begin{array}{lll} \underline{\underline{a}} \sim \underline{\underline{b}} = \underline{\underline{g1}} & (85a) ; & \underline{\underline{c}} \sim \underline{\underline{h}} = \underline{\underline{h}} & (85b) \\ \underline{\underline{y1}} \leftarrow \begin{smallmatrix} \text{NAN} \\ \sim \sim \end{smallmatrix} \underline{\underline{z}} = \underline{\underline{g2}} & (85c) ; & \underline{\underline{g1}} \vee \underline{\underline{g2}} = \underline{\underline{g2'}} & (85d) \\ \underline{\underline{x}} \sim \underline{\underline{j}} = \underline{\underline{g3}} & (85e) ; & \underline{\underline{g3}} \vee \underline{\underline{g2'}} = \underline{\underline{g}} & (85f) \\ \underline{\underline{g}} \sim \underline{\underline{h}} = \underline{\underline{k}} & (85g) ; & \underline{\underline{g}} \text{ ON } \underline{\underline{k}} = \underline{\underline{y2}} & (85h) \\ \underline{\underline{y1}} \vee \underline{\underline{y2}} = \underline{\underline{y}} & (85i) \end{array}$$

It is to be noted that the Eqns (85a-i) are those that are obtained after performing Steps B and C of the Flow Chart in Fig.1. Now, we perform the Steps D, E, F of the Flow Chart.

Step D: c-reconstruction of binary reverse relation

There is only one binary reverse relation in the argument under study —namely (85c). If we put $\underline{\underline{y1}} = T$ in this, the c-reconstruction yields the equations:

$$\underline{\underline{y1}} = T \quad (86a) ; \quad \underline{\underline{z}} \begin{smallmatrix} \text{NAN} \\ \sim \sim \end{smallmatrix} \underline{\underline{g2}} = T \quad (86b)$$

If, on the other hand, we take the case of $\underline{\underline{y1}} = F$, the c-reconstruction yields the equations:

$$\underline{\underline{y1}} = F \quad (87a) ; \quad \underline{\underline{z}} \begin{smallmatrix} \text{NI} \\ \sim \sim \end{smallmatrix} = \underline{\underline{g2}} \quad (87b)$$

Of these, (87b) corresponds to (80b) of the simple argument discussed in Subsection 4(c)(i) above and therefore we consider this first. It is illustrated in Fig. 4(b), where the change from Fig. 4(a) produced by the c-reconstruction is indicated.

(Note that the matrix $|NI| = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \equiv |0|$ and its complement $|0^c| = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \equiv |N|A|N|$, an A-type matrix with one 1 and three zeros.)

If y₁ is taken to be T and the corresponding c-reconstruction is made, as in Eqn. (86b), then an A-construction has also to be made. Both these together are shown in Fig. 4(c). Since the A-type operator involved is NAN, namely "nand", we have the consequences z = F and g₂ = F. Of these, g₂ = F is shown straightaway in Fig. 4(c). We could also say that z = F is the only possible state of z, as it is an input (t^(ai)), with no edge leading to it. However, the A-construction is fully shown in Fig. 4(c) to take care of the general case where z may be a middle term t^(m). The relevant equations for y₁ = T in Fig. 4(a)

are therefore (putting $\underline{\underline{z}} = \underline{\underline{z}2}$) :

$$\underline{\underline{y}1} = T \quad (88a) ; \quad \underline{\underline{z}1} = F \quad (88b)$$

$$\underline{\underline{z}1} \vee \underline{\underline{z}2} = \underline{\underline{z}} \quad (88c) ; \quad \underline{\underline{z}2} = F \quad (88d)$$

In these, $\underline{\underline{z}} = X$ if $\underline{\underline{z}2} = T$, so that we are led back to the possibility of $\underline{\underline{z}}$ (of (88c)) being only F, as in (86b), if $\underline{\underline{y}1} = T$.

In the subsection 4(d)(iii), we shall show how Steps E and F of the Flow Chart are implemented for the two graphs we have obtained, namely those corresponding to Figs. 4(a) and (b), and to Figs. 4(a) and (c) respectively. In the following subsection 4(d) (i), we consider the solution of the equations corresponding to the simpler argument delineated in Fig. 3.

(d) Working out the outputs of logical graphs

(i) Solution of the reduced graph of the simpler problem.

As already mentioned, we are given that the original inputs $\underline{t}^{(i)}$ are $\underline{a} = T$, $\underline{b} = F$, $\underline{c} = T$, in the reduced graph (Fig. 3). We also have, as additional inputs

$\underline{t}^{(ai)}$, the terms \underline{x} and \underline{z} , which are sought to be determined.

Therefore, we solve Eqns (80a-f), with the assumed states T and F for both \underline{x} and \underline{z} (four possibilities), and, in each case, determine the state of \underline{y} . When this is done with Eqns. (80a) to (80f), we obtain the results shown in Table 2.

Table 2: Solution of Equations (80a-h) pertaining to Fig.3

It will be seen from Table 2 that the combinations ($\underline{x} = T, \underline{z} = F$) and ($\underline{x} = F, \underline{z} = F$) as additional inputs, lead to contradictions. Hence, the argument cannot lead to $\underline{z} = F$.

Taking $\underline{z} = T$, the "additional input" \underline{x} may be T or F. For both possibilities, there are no contradictions, and both lead to $\underline{y} = F$. Hence, \underline{x} is indefinite (D), $\underline{y} = F, \underline{z} = T$ for this argument, with $\underline{a} = T, \underline{b} = F, \underline{c} = T$. No definite conclusion — namely whether $x = T$ or F, can be deduced, with these inputs, for this argument.

Table 2 : Solution of Equations (80a-h) pertaining to Fig.3

Inputs $\underline{t}^{(i)}$: $\underline{a} = T$, $\underline{b} = F$, $\underline{c} = T$

$\underline{t}^{(ai)}$	$\underline{t}^{(1)}$				$\underline{t}^{(2)}$	$\underline{t}^{(3)}$	$\underline{t}^{(4)}$	$\underline{t}^{(5)}$	
\underline{x}	\underline{z}	$\underline{g1}$	$\underline{g2}$	$\underline{g3}$	\underline{h}	$\underline{g2'}$	\underline{g}	\underline{k}	\underline{y}
T	T	F	D	D	T	F	F	T	F
T	F	F	T	D	F	X	X	X	X
F	T	F	D	F	T	F	F	T	F
F	F	F	T	F	D	X	X	X	X

In fact, this can be shown very simply by inverting the two equations (79b) and (79d) and rewriting the six equations in (79a) to (79f) as in (89a) to (89f) below. (Note that $\underline{z} \rightarrow \underline{x}$ = \underline{y} is written as $\underline{z} \rightarrow \underline{x} = \underline{y}$ and $\underline{x} \rightarrow \underline{z} = \underline{y}$ is written as $\underline{x} \rightarrow \underline{z} = \underline{y}$).

$$\underline{a} \& \underline{b} = \underline{g} ; \quad \underline{a} = T, \quad \underline{b} = F, \quad \text{yields} \quad \underline{g} = F \quad (89a)$$

$$\underline{c} \Rightarrow \underline{x} ; \quad \underline{c} = F, \quad \text{yields} \quad \underline{x} = D \quad (89b)$$

$$\neg \underline{e} \Rightarrow \underline{z} ; \quad \neg \underline{e} = F, \quad \text{yields} \quad \underline{z} = T \quad (89c)$$

$$\underline{c} \Rightarrow \underline{h} ; \quad \underline{c} = T, \quad \text{yields} \quad \underline{h} = T \quad (89d)$$

$$\underline{e} \vee \underline{h} = \underline{k} ; \quad \underline{e} = F, \quad \underline{h} = T, \quad \text{yields} \quad \underline{k} = T \quad (89e)$$

$$\underline{e} \vee \neg \underline{k} = \underline{y} ; \quad \underline{e} = F, \quad \underline{k} = T, \quad \text{yields} \quad \underline{y} = F \quad (89f)$$

The conclusion is that this argument, with inputs $\underline{a} = T$, $\underline{b} = F$, exactly $\underline{c} = F$ leads to $\underline{x} = D$, $\underline{y} = F$, $\underline{z} = T$, as deduced above, from Fig.3, via the steps in the Flow Chart.

Although such rewriting of the argument by the person solving it can appreciably simplify it, our purpose here is

mainly to show that there is a computerizable algorithmic procedure (as shown in the Flow Chart of Fig. 1) that can be given, and that it can solve for the states of the required (output) terms of any logical graph without human interference.

We have shown this for the problem composed of Eqs (79a-f).

We shall now obtain the reduced graphs and its equations for the two possibilities $\underline{y}_1 = T$ and F of the larger argument of of Fig.4(a).

(ii) Solution of the argument of the larger graph in Fig. 4(a)

The inputs ($\underline{t}^{(i)}$) to this graph are $\underline{a}, \underline{b}, \underline{c}$; additional inputs are $\underline{x}, \underline{z}, \underline{y}_1$ and the output \underline{y} . Suppose, as in Fig.3, we are given that $\underline{a} = T, \underline{b} = F, \underline{c} = T$. For the three additional inputs, we choose T and F for each of $\underline{x}, \underline{z}$ and \underline{y}_1 and get eight possibilities. We shall divide these into two sets, namely with $\underline{y}_1 = F$ and $\underline{y}_1 = T$, and consider them separately.

Case α : $\underline{\underline{y}}_1 = F$ corresponds to Figs 4(a) and 4(b).

85(a-h) with 87(a,b)

In this case, the implementable equations/are identical with those of the simpler argument in Fig.3, namely Eqs. (81a) to (81h), except for the following change and addition at the end of that list.

Change (81h) to $\underline{\underline{g}} \text{ ON } \underline{\underline{k}} = \underline{\underline{y}}_2$ (90)

$\underline{\underline{y}}_1 = F$ (91a) ; $\underline{\underline{y}}_1 \vee \underline{\underline{y}}_2 = \underline{\underline{y}}$ (91b)

Hence, Table 2 holds good, with $\underline{\underline{y}}(t^{(5)})$ replaced by $\underline{\underline{y}}_2$. We found therein that $\underline{\underline{y}}$ can only be F, so that, in the present case, $\underline{\underline{y}}_2 = F$. Putting this and the value $\underline{\underline{y}}_1 = F$ as in (91a) and (91b), we obtain that $\underline{\underline{y}} = F$. Hence, for case (α), the solution of the larger graph in Fig. 4(a) is

$\underline{\underline{x}} = D, \underline{\underline{y}} = F, \underline{\underline{z}} = T$ (92)

the same as for the simpler graph of Fig. 3.

We shall now consider the case when $\underline{\underline{y}}_1 = T$ in Fig. 4(a).

Case β : $\underline{y}_1 = T$, corresponding to Figs. 4(a) and 4(c)

In this case, we have chosen $\underline{y}_1 = T$, which makes (85c) become

$\underline{z}_1 \text{ NAN } \underline{g}_2 = T$, so that $\underline{z}_1 = F$ and $\underline{g}_2 = F$, as shown in

Fig.4(c). There is an independent disconnected graph $\underline{z}_1 \vee \underline{z}_2 = \underline{z}$

with $\underline{z}_1 = F$, so that the only non-contradictory solution for

\underline{z} is F , when $\underline{z}_1 = F$. Hence, we obtain for the equations for

the main reduced graph the same list as in (81a) to (81h), but

with the following changes and additions:

Change (81d) to $\underline{g}_2 = F$ (93a)

Change (81h) to $\underline{g} \text{ ON } \underline{b} = \underline{y}_2$ (93b)

Add the equations

$$\underline{y}_1 = T \quad (93c); \quad \underline{y}_1 \vee \underline{y}_2 = \underline{y} \quad (93d)$$

Add $\underline{z} = F$ from the disconnected graph (93e)

Hence, the only additional input is \underline{x} , which is put T and F

for the two possibilities. The consequences, on solving Eqs.

(81) and (93), are given in Table 3 below. It will be seen from

Table 3. Solution of Equations (80a-h) and (93a-e) of the larger graph-Case β .

Table 3: Solution of Equations (80a-h) and (95a-e) of
the larger graph — Case β

Inputs $\underline{t}^{(i)}$: $\underline{\underline{a}} = T$, $\underline{\underline{b}} = F$, $\underline{\underline{c}} = T$

$\underline{t}^{(ai)}$	$\underline{t}^{(1)}$	$\underline{t}^{(2)}$	$\underline{t}^{(3)}$	$\underline{t}^{(4)}$	$\underline{t}^{(5)}$
$\underline{\underline{x}}$ $\underline{\underline{z}}$ $\underline{\underline{g2}}$ $\underline{\underline{y1}}$	$\underline{\underline{g1}}$ $\underline{\underline{g2'}}$ $\underline{\underline{g3}}$ $\underline{\underline{h}}$	$\underline{\underline{g}}$	$\underline{\underline{k}}$	$\underline{\underline{y2}}$	$\underline{\underline{y}}$
T F F T	F F D T	- F	T	F	X
F F F T	F F F T	F	T	F	X

the table that the input state T of $\underline{\underline{y_1}}$ leads to the output

$\underline{\underline{y_2}} = T$, so that $\underline{\underline{y_1}} \vee \underline{\underline{y_2}} = \underline{\underline{y}}$ leads to a contradiction. This

happens for both the possibilities, $\underline{x} = T$ and $\underline{x} = F$. Hence,

there is no valid solution of the equations for this case

(corresponding to $\underline{\underline{y_1}} = T$) and the choice $\underline{\underline{y_1}} = T$ is impermissible.

Hence, for the graph in Fig. 4(a), $\underline{\underline{y_1}}$ can only be F ,

corresponding to case β , and this leads the solution in Eqs.(92),

namely $\underline{x} = D$, $\underline{y} = F$, $\underline{z} = T$, which are the same as those for

the simpler graph of Fig. 3.

Thus, we have considered the various types of constructions for obtaining the solution of logical graphs — such as c -construction A -reconstruction and V -construction for an SNS argument. The principles of these are unchanged when applied to QPL-1, QPL-2 and QSN statements forming parts of logical arguments, and are not particularly worked out. Instead, we consider an argument containing all these types of statements (but without the above requirements) in order to illustrate how these are tackled via our vector-matrix formalism.

(e) Example of an argument containing three types of terms

In this subsection, we shall give an example of the implementation of a logical graph (argument) containing QPL-1, QPL-2 and QSM relations. Since the main purpose is to illustrate the way in which these are worked out using the formalism developed by us, complexities of graph construction (as in Fig.3 and Fig.4 of the previous subsection) are avoided and no c-construction or A-reconstruction is required for this graph. On the other hand, interconversions of terms from QPL-1 to QPL-2 form and vice-versa, formulation and use of 3×3 and 3×2 matrices, theoretical combination of two QPL-1 statements to obtain a single resultant QPL-1 relation, working out of a unary bouquet etc., are illustrated. The argument, given in terms of logical equations in standard terminology, is converted into a sequentially implementable set of equations in our formalism, which is then implemented and the outputs worked out. The argument, which is rather complicated in order to illustrate all these possible intricacies / may be given in words as follows. (This argument, written

mainly to illustrate the above logical points, may not be a satisfactory paragraph in sociology.)

- (a) If there are strong leaders, then all people are restrained.
- (b) For all people, restraint and safety mean contentment.
- (c) For all people, discontent means dissatisfaction.
- (d) For some people, satisfied implies happy.
- (e) If some people are happy, then there is prosperity.
- (f) If all people are not happy, then there is trouble in all places, and vice-versa (i.e. if some people are happy, then there is no trouble in some places.)
- (g) If there exist leaders who are not corrupt, or there are people who fight for a pure administration, then, in all places, there will be no injustice.
- (h) If there exist criminal leaders, there will be injustice in all places.
- (i) If there is no trouble in all places, or there is no injustice in all places, then there is quiet peace; otherwise no quietude.

We give the logical equations for each of the statements

- (a) to (i) in (94a) to (94i) at the beginning of the next subsection (i). \therefore employ three variables:
 - (x) for leaders, (y) for people and (z) for places.

We are given the inputs:

All leaders are strong : $\alpha(x) = (\forall x)(a_T)$, in (a)

For all people there is safety: $b(y) = (\forall y)(b_T)$, in (b)

All leaders are not corrupt : $c(x) = (\forall x)(c_F)$, in (e)

There exist people who fight for a pure administration : $d(y) = (\exists y)(d_T)$, in (g)

We are required to find out the logical states of the

following outputs:

Prosperity : p (SNS term) in (e)

Quietude : q (SNS term) in (i)

Criminality of leaders: γ (QPL-1 term) in (h)

It is to be noted that the output γ in (94h) is put in as a QPL additional input $\underline{r}(\underline{t}^{(ai)})$, which is clear from Fig. 5 (See subsection (ii) below).

The other terms are denoted by the following symbols:

Restraint, of people : $\mathcal{C}(y)$ in (a)

Contentment, of people : $f(y)$ in (b)

Satisfaction, of people : $g(y)$ in (c)

Happiness, of people : $h(y)$ in (d)

Trouble, in places : $i(z) \equiv j(z)$ in (f)

Injustice, in places : $k(z) \equiv l(z)$ in (g), (h).

We now proceed to write the above argument in logical symbolism.

(i) Logical equations of the argument in standard language

The set of equations, in the standard terminology and symbology of predicate logic, is given below in (94a) to (94i).

$$(\exists x) (\underline{\underline{a}}(x)) \Rightarrow (\forall y) (\underline{\underline{e}}(y)) \quad (94a)$$

$$(\forall y) (\underline{\underline{e}}(y) \& \underline{\underline{b}}(y) = \underline{\underline{f}}(y)) \quad (94b)$$

$$(\forall y) (\neg \underline{\underline{f}}(y) \Rightarrow \neg \underline{\underline{g}}(y)) \quad (94c)$$

$$(\exists y) (\underline{\underline{g}}(y) \& \underline{\underline{h}}(y)) \quad (94d)$$

$$(\exists y) (\underline{\underline{h}}(y)) \Rightarrow \underline{\underline{p}} \quad (94e)$$

$$(\forall y) (\neg \underline{\underline{h}}(y)) \equiv (\forall z) (\underline{\underline{i}}(z)) \quad (94f)$$

$$(\exists x) (\neg \underline{\underline{c}}(x)) \vee (\exists y) (\underline{\underline{d}}(y)) \Rightarrow (\forall z) (\neg \underline{\underline{\ell}}(z)) \quad (94g)$$

$$(\exists x) (\underline{\underline{r}}(x)) \Rightarrow (\forall z) (\underline{\underline{\ell}}(z)) \quad (94h)$$

$$(\forall z) (\neg \underline{\underline{i}}(z)) \vee (\forall z) (\neg \underline{\underline{\ell}}(z)) = \underline{\underline{q}} \quad (94i)$$

The inputs are $\underline{\underline{a}}$, $\underline{\underline{b}}$, $\underline{\underline{c}}$, $\underline{\underline{d}}$ and the outputs are $\underline{\underline{p}}$, $\underline{\underline{q}}$ and

$\underline{\underline{M}}$ (or $\underline{\underline{r}}$). Of the three outputs, one (namely $\underline{\underline{M}}$) is a QPL-2

additional input in Eqn (94h). Since QPL terms are freely being converted from QPL-1 to QPL-2 and vice versa, we note that these

can be done by using the operators $\underline{\underline{C}}$ (canonizer) and $\underline{\underline{S}}$ (standardizer) for this purpose. In the logical graph, these will appear as unary operators. In Fig. 5 below, these alone are shown as small squares, while other logical connectives are shown as large squares, with the notation in our formalism — see subsection (ii) below.

Eqns.(94a) to (94i) are implementable, but for the following considerations:

Firstly, QPL-1 to QPL-2 form and QPL-1 to QPL-2 form should be expressed via canonizer and standardizer, in the form (in general) of $\mathcal{Q} \underline{\underline{C}} = \underline{\underline{g}}$ and $\underline{\underline{g}} \underline{\underline{S}} = \mathcal{Q}$ respectively.

Secondly, (1c) and (1d) are two QPL-1 relations in sequence and they must be combined theoretically. They have the pattern of Dimension (S1.1o.4. in Table 5 of Section 4(f)(i) of Part IV), with $\overline{\mathcal{N}}_f$ corresponding to $\underline{\underline{g}}$, $\underline{\underline{g}}$ corresponding to $\underline{\underline{b}}$ and $\underline{\underline{h}}$ corresponding

to \underline{c} of that Table 5, and hence, we obtain a single resultant QPL-1 relation, as in (95c) below:

$$\text{Replace (94c) and (94d) by } (\exists y) (\underline{f}(y) \Rightarrow \underline{h}(y)), \quad (95c)$$

Thirdly, as mentioned earlier in Section 2(d)(i), we must write (94g) as two equations (95g'), (95g'') as follows:

$$(\exists x) (\neg \underline{c}(x)) \vee (\exists y) (\underline{d}(y)) = \underline{k} \quad (\text{QPL-2 to SIS}) \quad (95g')$$

$$\underline{k} \Rightarrow (\forall z) (\neg \underline{\ell}(z)) \quad (\text{SIS to QPL-2}) \quad (95g'')$$

Fourthly, Eqns (95g'') and (94h) are found to have multiple inputs for $\underline{\ell}(z)$. Hence, we call the output of (95g'') as $\underline{\underline{\ell}}_1(z)$ and that of (94h) as $\underline{\underline{\ell}}_2(z)$, and add the equation (95h') as below:

$$\underline{\underline{\ell}}_1(z) \vee \underline{\underline{\ell}}_2(z) = \underline{\ell}(z) \quad (95h')$$

and $\underline{\ell}(z)$ is then input to (94i).

Eqns (94), with the modifications in (95), are implementable in sequence. However, while converting these to our language, we write them in the sequence that would be obtained using the algorithm, discussed in Section 4(c), for working out the argument from Fig.3. In this, a unary sequence (or bouquet) is worked out in one step.

(ii) Logical graph of the argument and equations in our language

The logical graph of the above argument (after making the modifications mentioned in subsection (i) above) is given in Fig.5. It will be seen from the graph that the initial inputs ($\underline{t}^{(i)}$) are a , b , c and d , whose states are as given in Table 4 below, and the outputs are p and q . The other input \underline{r}

Table 4. Working out the logical graph in Fig. 5 (pages 98 to 101)

is really an output, put in as additional input ($\underline{t}^{(ai)}$) and its equivalent QPL-1 form is \mathcal{N} , obtained by the application of the standardizer S . In view of this, we write the equation

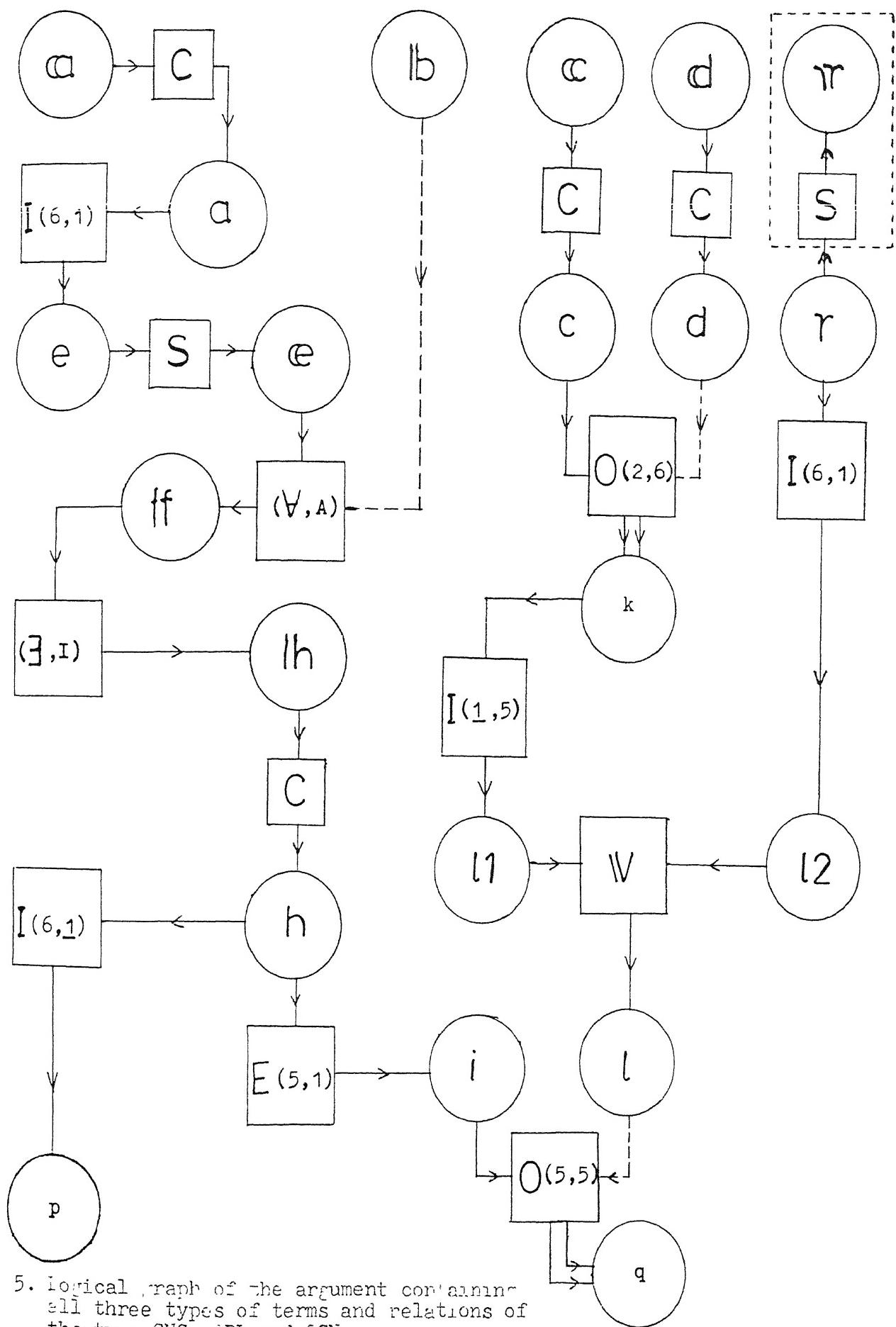
$$\underline{r} \quad \underline{S} = \mathcal{N} \quad (96)$$

and put it in a box in Fig.5 separately and input for \underline{r} ($\underline{t}^{(ai)}$) the three QPL-2 pure states (1 0 0), (0 1 0) and (0 0 1) as the three possibilities of this term \underline{r} .

For the rest of the graph, we follow the algorithmic procedure

Fig.5. Logical graph of the argument containing all three types of terms, and relations of the type SMS, QPL and QSN (in p 95a)

.95a.



5. Logical graph of the argument containing all three types of terms and relations of the type SNS, GPL and QSN

of starting from the $\underline{t}^{(i)}$'s and the $\underline{t}^{(ai)}$ and proceeding

step by step, as described in Section (d) above. Then, we get

the results as indicated in Table 4 where the steps of the

algorithm are also indicated. Since \underline{r} enters the picture

only as Eqn (97k), the equations from (97a) to (97j) are worked out

for the given states of the inputs $\underline{t}^{(i)}$, and the remaining

four equations (97k) to (97o) are alone worked out for the

three possibilities of \underline{r} .

As mentioned in Section 2, the symbology for a unary and binary Type-1 operators will be written as in (98a,b,c), so that it has the same structure as for QPL- Type-2 and SIS operators:

$$Q(\varphi, z) = \underline{b} \text{ stands for } Q(\underline{a} z = \underline{b}) \quad (98a)$$

$$Q(\varphi, z) \underline{b} = \underline{c} \text{ stands for } Q(\underline{a} z \underline{b} = \underline{c}) \quad (98b)$$

$$\underline{c} (\varphi, \underline{z}) a = \underline{b} \text{ stands for } Q(\underline{c} \underline{z} \underline{a} = \underline{b}) \quad (98c)$$

Also, the equivalence operator $\underline{\approx}(i, j)$ stands for the following:

$$(\underline{a} \underline{\approx}(i, j) = \underline{b}) \equiv (\underline{a} \underline{\approx}(i, j) = \underline{b}) \& (\underline{a} \underline{\approx}(i^c, j^c) = \underline{b}) \quad (99)$$

Also, in the matrix operators for QSN relations, the index

corresponding to the SNS term is underlined, to distinguish it

from a pure QPL-2 operator - e.g.

$$\underline{\underline{a}} \underline{\underline{I}}(\underline{i}, j) = \underline{\underline{b}}, \underline{i} = 1 \text{ to } 4, j = 1 \text{ to } 8, \text{ for SNS-QPL-2 relation (10Ca)}$$

$$\underline{\underline{a}} \underline{\underline{O}}(\underline{i}, \underline{j}) \underline{\underline{b}} = \underline{\underline{c}}, \underline{i} = 1 \text{ to } 8, \underline{j} = 1 \text{ to } 4, \text{ for QPL-2-SNS relation (10Cb)}$$

Coming back to the argument worked out in Table 4, we see

that it leads to the conclusions $\underline{\underline{p}} = T$, $\underline{\underline{q}} = T$, $\underline{\underline{r}} = (0 \ 0 \ 1)$ (F),

with the given inputs mentioned earlier. There is prosperity,

quietude, and no leaders are criminal. For obtaining the last

of these, we have used $\underline{\underline{r}}$ as an additional input in $\underline{\underline{r}} \underline{\underline{I}}(6, 1) = \underline{\underline{\ell}}_2$.

We can also obtain the same result by inverting this equation.

If we do this, there is no need to have $\underline{\underline{\ell}}_1$, $\underline{\underline{\ell}}_2$ for $\underline{\underline{\ell}}$.

Instead, we obtain, from (97m), the equation $\underline{\underline{k}} \underline{\underline{I}}(1, 5) = \underline{\underline{\ell}}$,

which, with $\underline{\underline{k}} = (1 \ 0)$, leads to $\underline{\underline{\ell}} = (0 \ 0 \ 1)$. The inverted

equation for $\underline{\underline{r}}$, obtained from (97k) is

$$\underline{\underline{\ell}} \underline{\underline{I}}(2, 5) = \underline{\underline{r}} \quad (101a)$$

Table 4. Working out the logical graph in Fig. 5*

Nature	Equation	Input(s)	Output	Eqn.No.
Initial inputs ($\underline{t}^{(i)}$) : $A = \forall(T)$, $B = \forall(T)$, $C = \forall(F)$, $D = \exists(T)$				
Additional inputs ($\underline{t}^{(ai)}$, : $\underline{\underline{x}} = (1\ 0\ 0)$, $(0\ 1\ 0)$ or $(0\ 0\ 1)$				
<u>Step 1</u>				
(QPL-1 to 2)	$\underline{A} \underline{\underline{C}} = \underline{\underline{a}}$	$\forall(T)$	$(1\ 0\ 0)$	(97a)
(QPL-2)	$\underline{\underline{a}} \underline{\underline{I}}(6,1) = \underline{\underline{e}}$	$(1\ 0\ 0)$	$(1\ 0\ 0)$	(97b)
(QPL-2 to 1)	$\underline{\underline{e}} \underline{\underline{S}} = \underline{\underline{C}}$	$(1\ 0\ 0)$	$\forall(T)$	(97c)
(QPL-1)	$\underline{\underline{C}} (\forall, A) \underline{B} = \underline{\underline{f}}$	$\forall(T), \forall(T)$	$\forall(T)$	(97d)
(QPL-1)	$\underline{\underline{f}} (\exists, I) = \underline{\underline{h}}$	$\forall(T)$	$\exists(T)$	(97e)
(QPL-1 to 2)	$\underline{\underline{h}} \underline{\underline{C}} = \underline{\underline{h}}$	$\exists(T)$	$(1\ 1\ 0)$	(97f)
(QPL-2 to SNS)	$\underline{\underline{h}} \underline{\underline{I}}(6,1) = \underline{\underline{p}}$	$(1\ 1\ 0)$	$(1\ 0)$	(97g)
(QPL-2)	$\underline{\underline{h}} \underline{\underline{S}}(5,1) = \underline{\underline{i}}$	$(1\ 1\ 0)$	$(0\ 1\ 1)$	(97h)
(QPL-1 to 2)	$\underline{C} \underline{\underline{C}} = \underline{\underline{c}}$	$\forall(T)$	$(1\ 0\ 0)$	(97i)
(QPL-1 to 2)	$\underline{D} \underline{\underline{C}} = \underline{\underline{d}}$	$\exists(T)$	$(1\ 1\ 0)$	(97j)
(QPL-2)	$\underline{\underline{r}} \underline{\underline{I}}(6,1) = \underline{\underline{f}}_2$	$(1\ 0\ 0)$ $(0\ 1\ 0)$ $(0\ 0\ 1)$	$(1\ 0\ 0)$ $(1\ 0\ 0)$ $(1\ 1\ 1)$	(97k)
Outputs of step 1 : $\underline{\underline{p}}, \underline{\underline{i}}, \underline{\underline{c}}, \underline{\underline{d}}, \underline{\underline{f}}_2$				

Table.4 (Contd)

Nature	Equation	Input(s)	Output	Eqn.No.
<u>Step 2</u>				
(QPL-2 to SNS)	$\underline{\underline{c}} \approx \underline{\underline{0}}(2,6) \underline{\underline{d}} = \underline{\underline{k}}$	(0 0 1), (1 1 0)	(1 0)	(97e)
Output of Step 2 : $\underline{\underline{k}}$				
<u>Step 3</u>				
(SNS to QPL-2)	$\underline{\underline{k}} \underline{\underline{I}}(1,5) = \underline{\underline{\ell}}_1$	(1 0)	(0 0 1)	(97m)
Output of Step 3 : $\underline{\underline{\ell}}_1$				
<u>Step 4</u>				
(QPL-2)	$\underline{\underline{\ell}}_1 \vee \underline{\underline{\ell}}_2 = \underline{\underline{\ell}}$	(0 0 1), (1 0 0) (0 0 1), (1 0 0) (0 0 1), (1 1 1)	(0 0 0) (0 0 0) (0 0 1)	(97n)
Output of Step 4 : $\underline{\underline{\ell}}$				
<u>Step 5</u>	$\underline{\underline{i}} \approx \underline{\underline{0}}(5,5) \underline{\underline{\ell}} = \underline{\underline{q}}$	(0 1 1), (0 0 0) (0 1 1), (0 0 0) (0 1 1), (0 0 1)	(0 0) (0 0) (1 0)	(97o)
Output of Step 5 : $\underline{\underline{q}}$				
Conversion of $\underline{\underline{r}}$ to QPL-1 from \mathcal{M} (for non-contradictory $\underline{\underline{r}}$)				
(QPL-2 to QPL-1)	$\underline{\underline{r}} \approx \underline{\underline{s}} = \mathcal{M}$	(0 0 1)	$\forall(F)$	(97p)

* The details of working out the equations (97a) to (97p) are given in the notes below.

(97a) : Using the canonizer in Table 7, Part II,

$$\mathcal{A} = \forall(T) \xrightarrow{C} \underline{\underline{a}} = (1 \ 0 \ 0)$$

(97b) : Using the theory in Section 5(f), Part II, we obtain

$$(1 \ 0 \ 0) \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix} = (1 \ 0 \ 0)$$

(97c) : Using the standardizer in Table 7, Part II, suitably modified, we obtain

$$(1 \ 0 \ 0) \xrightarrow{S} \forall(T)$$

(97d) : Using Eq.(32c) in Section 3(c) (i) of Part IV, we have

$\mathcal{V}'_e = \forall$, $\underline{e} = T$; $\mathcal{V}'_b = \forall$, $\underline{b} = T$ in $\forall(\underline{e} \wedge \underline{b} = \underline{f})$, and we obtain

$$\mathcal{V}'_f = \mathcal{V}'_c \oplus \mathcal{V}'_b = \forall, \underline{f} = \underline{e} \wedge \underline{b} = T, \text{ giving } \underline{f} = \forall(T)$$

(97e) : Using Fig. 6(a) of Part IV, we have $\mathcal{V}'_f = \forall$, $\underline{f} = T$ in $\exists(\underline{f} \underline{I} = \underline{h})$, so that $\mathcal{V}'_h = (\forall \oplus \exists) \oplus \exists = \exists$ and $\underline{h}' = \underline{f} \underline{I} = T$, giving $\underline{h} = \exists(T)$.

(97f) : As in (97a), we use Table 7, Part II to give $\exists(T) \xrightarrow{C} (1 \ 1)$

(97g) : As in (97b), we obtain, for the 3X2 matrix $\underline{\underline{I}}(6, 1)$,

$$(1 \ 1 \ 0) \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 1 \end{pmatrix} = (1 \ 0) = T$$

(97h) : We use the result $(\underline{h} \ \underline{E}(5, 1) = \underline{i}) \equiv ((\underline{h} \ \underline{I}(5, 1) = \underline{i}) \ \& \ (\underline{h} \ \underline{I}(6, 2) = \underline{i}))$, so that

$$\underline{E}(5, 1) = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix} \otimes \begin{pmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

and with $\underline{h} = (1 \ 1 \ 0)$, $\underline{i} = (1 \ 1 \ 0) \begin{pmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix} = (0 \ 1 \ 1)$

(97i), (97j) : Both these use the canonizer $\underline{\underline{G}}$

(97k) : Using Section 6(f) of Part II, we have

$$(1 \ 0 \ 0) \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix} = (1 \ 0 \ 0); \quad (0 \ 1 \ 0) \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix} = (1 \ 0 \ 0);$$

$$(0 \ 0 \ 1) \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix} = (1 \ 1 \ 1)$$

(97l) Using the theory in Part II, with $\underline{\underline{C}} = (0 \ 0 \ 1)$

$$(0 \ 0 \ 1) \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} = (1 \ 1 \ 1) \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = 1 = k_{\alpha}$$

$$(0 \ 0 \ 1) \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} = (0 \ 0 \ 0) \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} = 0 = k_{\beta}$$

so that $\underline{\underline{k}} = (k_{\alpha} \ k_{\beta}) = (1 \ 0) = T$.

(97m) : As in (97g), we use the 2×3 matrix for $\underline{\underline{I}}(1, 5)$ and obtain

$$(1 \ 0) \begin{pmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix} = (0 \ 0 \ 1)$$

(97n) : For the three possibilities for $\underline{\underline{\ell}}_2$ in (97k), we obtain

$$(0 \ 0 \ 1) \otimes (1 \ 0 \ 0) = (0 \ 0 \ 0); \quad (0 \ 0 \ 1) \otimes (0 \ 1 \ 0) = (0 \ 0 \ 0);$$

$$(0 \ 0 \ 1) \otimes (0 \ 0 \ 1) = (0 \ 0 \ 1)$$

(97o) : As in (97b), remembering that the impossible state $\phi = (0 \ 0 \ 0)$ can lead only to $X = (0 \ 0)$, we obtain for the only non- ϕ state of $\underline{\underline{\ell}}_2$, namely $(0 \ 0 \ 1)$,

$$(0 \ 1 \ 1) \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = (1 \ 1 \ 1) \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = 1 = q_{\alpha}$$

$$(0 \ 1 \ 1) \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = (1 \ 1 \ 0) \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = 0 = q_{\beta}$$

so that the output $\underline{\underline{q}} = (q_{\alpha} \ q_{\beta}) = (1 \ 0) = T$.

(97p) : Using the standardiser, as in (97c), $(0 \ 0 \ 1) \xrightarrow{S} \forall(\bar{x})$

which yields

$$\underline{\underline{r}} = (0 \ 0 \ 1) \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix} = (0 \ 0 \ 1) \quad (101b)$$

This is the same as what we have obtained for $\underline{\underline{r}}$ in Table 4,

assuming the three basic states of IA-3 for it, and making the consistency check in each case. As mentioned earlier, we shall prove in Section 5, this property of the inversion of a statement, for all implications and equivalences in GFL-2, SIS and QSM, from the property of $\underset{\approx}{I}(i, j)$ that it is equivalent to $\underset{\approx}{I}(j^c, i^c)$ in the inverted relation.

Thus we have considered practical examples of a logical graph in SIS having various complications in the logical interconnections occurring in the graph and also an example of a simple logical graph, but in which all the different types of relations such as RPI, SNS, ISI occur. By combining the features which occur in examples considered both in Section 4(d) and 4(e) above, it should be possible to take care of

practically any logical argument with terms which are logical functions of only one variable.

In the next section, we shall consider some more general relations and their interpretation in our language in terms of suitable matrices, either 3×3 , 2×2 , 2×3 or 3×2 . We shall not consider in this report arguments having terms which are logical functions of more than one variable such as $\underline{\underline{a}}(x, y)$. It should, however, be recognized that the unary and binary connective operators could be functions of one variable or two variables, in the various examples considered above. Thus, as mentioned in Eq.(20) of Section 2(b), the simple relation $\underline{\underline{a}}(x) \approx (i, j) = \underline{\underline{b}}(y)$ really has for the implication operator the form $\approx (i, x; j, y)$ where the first variable x is to be connected with the input term $\underline{\underline{a}}(x)$ and the second variable y of the operator is to be associated with the variable of the output $\underline{\underline{b}}(y)$. Such features will be discussed further when we consider more general logical relations in which the terms themselves may be functions of more than one variable.

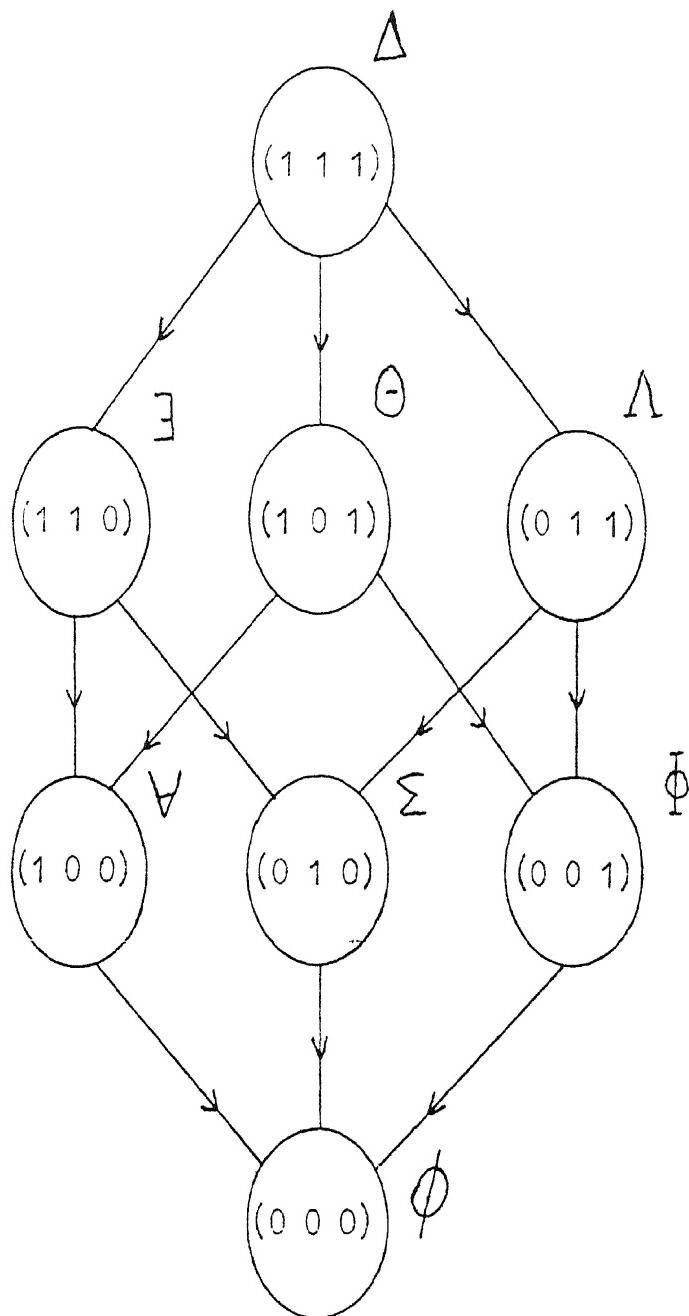
5. Some theorems and special considerations

First we shall consider the proof of the fact that the inversion of an implication checked via the consistency operator ∇ and the direct application of the implication in the forward sense will both lead to the same result. We saw particular examples of this in Section 4(b) and 4(e)(ii). We shall give a general proof first for QPL-2 unary implications $\tilde{\approx}(i, j)$ and then indicate how it is equally valid for SNS and QSN relations of the same type.

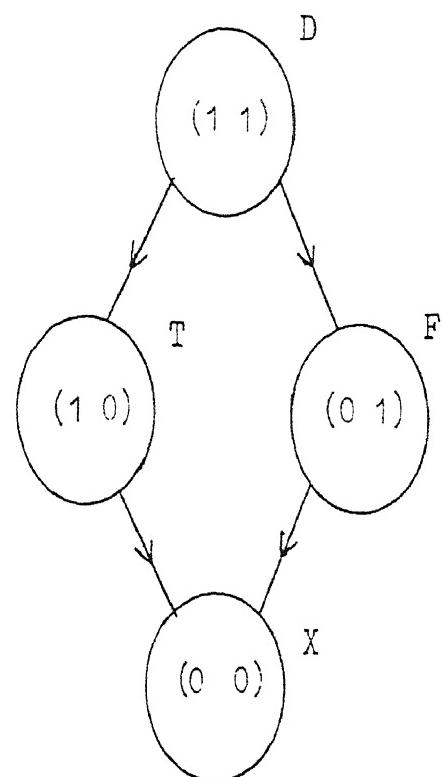
(a) Inversion of an implication considered via consistency checks.

First we shall indicate the well-known lattice-type relationships between $q(1)$ to $q(8)$ of QPL and also between $s(1)$ to $s(4)$ of SNS. These are shown in Fig. 6(a) and (b).

Fig. 6. Inclusion relations shown by lines with arrows for
(a) QPL states represented by 3-element Boolean vectors
of BA-3, and (b) SNS states represented by 2-element
Boolean vectors of BA-2 .



(a)



(b)

Fig.6 Inclusion relations shown by lines with arrows for
 (a) QFL states represented by 3-element Boolean
 vectors of EA-3, and (b) SFS states represented
 by 2-element Boolean vectors of EA-2.

In this figure a state $q(j)$, connected by a directed line to $q(k)$ below it, means that the vector $(\gamma_j \ \delta_j \ \epsilon_j)$ representing the former contains in it all the one's of the vector $(\gamma_k \ \delta_k \ \epsilon_k)$ associated with $q(k)$. We shall represent it by the symbology:

$$\text{Lattice inclusion} : q(j) \subset q(k) \quad (102)$$

We wish to prove the result that if

$$\underline{\underline{a}} \underset{\sim}{\approx} (i, j) = \underline{\underline{b}} \quad (103)$$

then, in the inverse relation between $\underline{\underline{b}}$ and $\underline{\underline{a}}$ represented by

Eqs (104a and b),

$$\underline{\underline{b}} \underset{\sim}{\approx} (j^c, i^c) = \underline{\underline{a}}' , \quad \underline{\underline{a}} \vee \underline{\underline{a}}' \neq \emptyset \quad (104a, b)$$

if $\underline{\underline{b}}$ is given successively the three basic states $q(1)$ ($\equiv (1 0 0)$), $q(3)$ ($\equiv (0 1 0)$), and $q(5)$ ($\equiv (0 0 1)$, of QPL, and those that do not lead to the null set \emptyset in (104b) are added together, then $\underline{\underline{b}}$ so obtained will be the same as in (103).

The proof follows straightforwardly from the fact that the implication (103) means that, only if $\underline{\underline{a}} \subset q(i)$, will $\underline{\underline{b}}$ be equal

to $\underline{q}(j)$. Then if we take $\underline{\underline{b}}' \subseteq \underline{q}^c(j) \equiv q(j^c)$, then, by (104a),

$$\underline{\underline{b}}' \mapsto \underline{\underline{a}}' \subseteq \underline{q}(i^c) \equiv q^c(i) \quad (105)$$

Consequently $\underline{\underline{a}}' \otimes \underline{\underline{a}} = q^c(i) \otimes q(i) = \emptyset$ for $\underline{\underline{a}}'$ given by the inverted implication. On the other hand, if $\underline{\underline{b}}' \subseteq q(j)$ is input in (104a), it leads to $\underline{\underline{a}}' = (1 1 1) = \Delta$, and this yields

$\underline{\underline{a}}' \vee \underline{\underline{a}} \neq \emptyset$. Hence, the consistency check gives a non-impossible result only if $\underline{\underline{b}}' \subseteq q(j)$. This means that the state of the term $\underline{\underline{b}}$ obtained by the consistency check is $q(j)$, the same as that obtained by the forward relation (103) when $\underline{\underline{a}} \subseteq q(i)$.

This proof for QPL is equally valid for SNS without any change except that the 3-vectors $q(j)$, $q(k)$ are replaced by $s(j)$, $s(k)$ which are 2-vectors, and Eq. (105) is replaced by the SNS relation $\underline{\underline{a}} \underset{I}{\sim} \underline{\underline{b}}$. We then obtain the same conclusion, namely that this direct implication relation gives the same result as that obtained from the inverted implication relation $(\underline{\underline{b}} N) \underset{I}{\sim} (\underline{\underline{a}} N)$ by checking for consistency. (Note that, in SNS, $(\underline{\underline{a}} N) = \underline{\underline{a}}^c$, for the basic states T and F).

By combining the properties of 3-vectors in QPL and 2-vectors in SLS, we can write a proof in an exactly similar manner for the LSL type of implications, namely, QPL-2 to SNS and SNS to QPL-2.

The essential equation used is $(\underline{\underline{a}} \underline{\underline{I}}(i, j) = \underline{\underline{b}}) \equiv (\underline{\underline{b}} \underline{\underline{I}}(\underline{\underline{j}}^c, \underline{\underline{i}}^c) = \underline{\underline{a}})$ for the former, and $(\underline{\underline{a}} \underline{\underline{I}}(i, j) = \underline{\underline{b}}) \equiv (\underline{\underline{b}} \underline{\underline{I}}(\underline{\underline{j}}^c, \underline{\underline{i}}^c) = \underline{\underline{a}})$ for the latter

The expansion of the above proof from $\underline{\underline{I}}(i, j)$ to $\underline{\underline{E}}(i, j)$ for QPL, where $\underline{\underline{E}}(i, j)$ satisfies the equivalence (99) is also obvious, since both the implications relations on the r.h.s of (99) lead to identical results, namely that

$$(\underline{\underline{a}} \underline{\underline{\approx}}(i, j) = \underline{\underline{b}}) \quad (\underline{\underline{b}} \underline{\underline{\approx}}(\underline{\underline{j}}^c, \underline{\underline{i}}^c) = \underline{\underline{a}}) \quad (106a)$$

$$(\underline{\underline{a}} \underline{\underline{\approx}}(\underline{\underline{i}}^c, \underline{\underline{j}}^c) = \underline{\underline{b}}) \quad (\underline{\underline{b}} \underline{\underline{\approx}}(\underline{\underline{j}}, \underline{\underline{i}}) = \underline{\underline{a}}) \quad (106b)$$

We shall not elaborate on this further.

(b) Other general matrix operators

We have considered matrix operators of the types $\underline{\underline{A}}(i, j)$, $\underline{\underline{G}}(i, j)$, $\underline{\underline{I}}(i, j)$, $\underline{\underline{J}}(i, j)$ and $\underline{\underline{E}}(i, j)$ so far. These do not, by any means, exhaust all the possible 3×3 Boolean matrices. The general type of 3×3 matrix appears to be complicated, but any

such general matrix can be expressed as a combination of two implications, or three implications, as shown below. One example of two implications was (49) where the equivalence operator $\mathcal{E}(i, j)$ was defined in terms of two implications. Another example of such a relation is the following in QSI:

$$\left[\underline{\underline{a}} = (1 \ 0 \ 0) \right] \Rightarrow \left[\underline{\underline{b}} = (1 \ 0) \right]; \quad \left[\underline{\underline{a}} = (0 \ 0 \ 1) \right] \Rightarrow \left[\underline{\underline{b}} = (0 \ 1) \right] \quad (107)$$

Written in standard terminology, these become

$$\forall(\underline{\underline{a}}) \Rightarrow b_T; \quad \forall(\neg \underline{\underline{a}}) \Rightarrow b_F \quad (108a, b)$$

Obviously, the third possibility namely $\underline{\underline{a}} = (0 \ 1 \ 0)$, i.e. $\underline{\underline{a}} \equiv \exists$, but not \forall , is left unspecified in (108a, b) and therefore we should add this in the form of Eq. (108c):

$$\left[\underline{\underline{a}} = (0 \ 1 \ 0) \right] \Rightarrow \left[\underline{\underline{b}} = (1 \ 1) \right], \quad \text{or} \quad \exists(\underline{\underline{a}}) \Rightarrow b_D \quad (108c)$$

The three implications in (108) can be represented by the vector-matrix relations

$$\langle a_1 | z_1 \rangle = \langle b_1 |, \quad \langle a_2 | z_2 \rangle = \langle b_2 |, \quad \langle a_3 | z_3 \rangle = \langle b_3 | \quad (109)$$

where

$$|z_1| = |z(1, 1)| = \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 1 \end{pmatrix}, \quad |z_2| = |z(3, 2)| = \begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{pmatrix},$$

$$|z_3| = |z(5, 2)| = \begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 0 & 1 \end{pmatrix} \quad (110)$$

.110.

if we combine these in the form of a QSN equation

$$\underline{\underline{a}} \underline{\underline{z}} = \underline{\underline{b}}, \text{ or } \langle \underline{\underline{a}} | \underline{\underline{z}} \rangle = \langle \underline{\underline{b}} | \quad (111a,b)$$

then obviously, the matrix of the operator $\underline{\underline{z}}$ will be

$$|\underline{\underline{z}}| = |\underline{\underline{I}}(1, 1)| \otimes |\underline{\underline{I}}(3, 2)| \otimes |\underline{\underline{I}}(5, 2)| \quad (112a)$$

and the 3×2 matrix $|\underline{\underline{z}}|$ has the form

$$|\underline{\underline{z}}| = \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{pmatrix} \quad (112b)$$

It is readily verified that this matrix has the properties mentioned above, by substituting it in (111b).

In this way, any general 3×3 , or 3×2 , matrix is capable of being written as the product of three implication matrices.

A 2×3 matrix can also be considered similarly as a product of two SIS-IPL implications.

For example, the 3×3 matrix $|\underline{\underline{z}}|$ of (113) below,

$$|\underline{\underline{z}}| = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \quad (113)$$

can be interpreted, for $\langle \underline{a} | z | = \langle \underline{b} |$, as

$$q_a(1) \mapsto (0 \ 0 \ 1) \equiv \forall(b_T); \quad q_a(3) \mapsto (1 \ 1 \ 0) \equiv \exists(b_T);$$

$$q_a(5) \mapsto (1 \ 0 \ 0) \equiv \forall(b_T) \quad (11 \cdot a, b, c)$$

which, in words take the form

- (a) If \underline{a} is true for all, then \underline{b} is true for none (\emptyset)
- (b) If \underline{a} is true for some, then \underline{b} is true for some or for all (\exists) (115)
- (c) If \underline{a} is true for none, then \underline{b} is true for all (\forall).

If \underline{a} has a mixed state, the state of \underline{b} is obtained simply by

summing, using the operator U , the outputs corresponding to

each of the basic states contained in \underline{a} . Thus, if $\underline{a} = \{ \} \equiv (1 \ 1 \ 0)$,

then, for the matrix $|z|$ of (113),

$$\underline{b} = (0 \ 0 \ 1) \oplus (1 \ 1 \ 0) = (1 \ 1 \ 1) \equiv \Delta, \text{ the indefinite state} \quad (116)$$

If, however, $\underline{a} = \Lambda = (0 \ 1 \ 1)$, we obtain a definite state for \underline{b}

as follows:

$$\underline{b} = (1 \ 1 \ 0) \oplus (1 \ 0 \ 0) = (1 \ 1 \ 0) \equiv \exists \quad (117)$$

We leave the further working out of these considerations for a

later report.

Acknowledgement

Parts III, IV and V of this series were written by the author as a CSIR Distinguished Scientist (Retd.) during January to June 1984. The work was supported by an Extramural Grant from CSIR, India, and the author is deeply grateful to Dr.G.S.Sidhu Director-General, CSIR, for having provided him with this grant as a special case. Thanks are due to Mr.N. Suresh for providing excellent secretarial assistance for the preparation of these reports.

List of References

1. Ranachandran, G.N., Vector-Matrix Representation of Boolean Algebras and Application to Extended Predicate Logic (EFL) Part I, Current Science, 1983, 52, 292-302.
2. -do- Part II, Current Science, 1983, 52, 335-341.
3. -do- Part III "Algebraic Theory for Categorical Statements in Quantified Predicate Logic (QPL)" (Submitted for publication) (Matphil Reports No. 34).
4. -do- Part IV "General Theory for Statements of Type-1 in EFL". (Matphil Reports No. 35, Draft 2.)
5. Ranachandran, G.N., Syad-Nyaya System (SNS) — A New Formulation of Sentential Logic and its Isomorphism with Boolean Algebra of Genus 2, Current Science, 1982, 51, 625-636.
6. Suppes, P., Introduction to Logic, Van Nostrand, New York, 1957, p.50.

FIGURES

	<u>Page No</u>
1. Flow Chart for the Implementation of a General Argument in Logic (for one variable)	50
2. Logical graph of a simple argument consisting of two steps	
(a) As given in Eq (72 a & b).	58
(b) The same after <u>V</u> -construction to remove multiple inputs.	58
(c) The same, after inverting the second Eq (72b) to obtain the pair of Eqs (73a, b).	58
3. Reduced graph of the argument composed of Eqs.(79), modified as in (80).	69
4. (a) Graph of the argument composed of Eqs. (79), (80) and (82), after the "circuit removal construction" in Step B, and the " <u>V</u> -construction" for removing multiple inputs in Step C, of the Flow Chart.	76
(b) <u>c</u> -reconstruction (Step D of Flow Chart) when <u>y1</u> = F. (See also Fig. 7(b) of Part IV.)	77
(c) <u>c</u> -reconstruction (as in Step D of Flow Chart) when <u>y1</u> = T and <u>A</u> -construction for <u>NAN</u> (as in Step F) to <u>z1</u> = F and <u>z2</u> = F. (See also Fig.6(b) of Part IV.)	77
5. Logical graph of the argument containing all three types of terms, and relations of the type SNS, QPL and QSN	95a
6. Inclusion relations shown by lines with arrows for (a) QPL states represented by 3-element Boolean vectors of BA-3, and (b) SMS states represented by 2-element Boolean vectors of BA-2.	105

TABLES

	<u>Page No.</u>
1. Checking for non-contradictory inputs for \leq in Fig. 2(b).	60
2. Solution of Equations (80a-h) pertaining to Fig. 3	82
3. Solution of Equations (80a-h) and (93a-e) of the larger graph- Case β .	87
4. Working out the logical graph in Fig. 5 (pages 98 to 101)	98

Studies in Mathematical Logic*

Part VI - Elimination of Paradoxes and Incompleteness from Logic

(*This is the continuation of the series "Vector-Matrix Representation of Boolean Algebras and Application to Extended Predicate Logic (EPL)" - Part I to V.)

G.N. Ramachandran

INSA Albert Einstein Professor
Mathematical Philosophy Group
Indian Institute of Science
Bangalore 560 055

Matphil Reports No. 37, August 1984

(Draft 1 - for private circulation)

Studies in Mathematical Logic*

Part VI - Elimination of Paradoxes and Incompleteness from Logic

(* This is the continuation of the series "Vector-Matrix Representation of Boolean Algebras and Application to Extended Predicate Logic (EPL)" — Parts I to V.)

G.N. Ramachandran

Gita, 5, 10A Main Road
Mallewaram West
Bangalore 560 055

Matphil Reports No. 37, August 1984
(Draft 1 — for private circulation)

Preface and Summary

This report deals with two important consequences of the logical representation of both sentential logic and predicate logic which we have developed, using Boolean algebras BA-2 and BA-3. These are that two defects necessarily present in the conventional formulations of logic, namely the existence of paradoxes, and the existence of incompleteness in predicate logic, can both be eliminated by taking into account the new "mixed" states that automatically come in when the complete set of Boolean algebraic states in each case is taken into account.

Thus, even propositional calculus (PC), having only the two mutually exclusive states T and F., is incomplete, in that the "mixed" state D (\equiv "may be true or may be false") is a necessary consequence of reversing standard connectives like "or" and "if" in PC itself. However, the incompleteness is removed when PC is extended to SNS, isomorphic to BA-2, and D becomes (1 1), with T = (1 0) and F(0 1). In the same way, using BA-3 representation of predicate logic, which has only the four states-

.ii.

"for all" (\forall), "not for all" (Λ), "for none" (\emptyset) and

"there exists" (\exists), it leads to the existence of the state

envisioned in Gödel's incompleteness theorem (which we denote by Σ)

by the application of the consistency check operator (∇) to

\exists and $\Lambda - \exists \nabla \Lambda = \Sigma$. Thus $\exists \otimes \Lambda = (1 \ 1 \ 0) \otimes (0 \ 1 \ 1) =$

$(0 \ 1 \ 0) = \Sigma$. This state Σ is not envisioned in standard

predicate logic, but is a necessarily present basic state $(0 \ 1 \ 0)$

in BA-3, in which $\forall \exists (1 \ 0 \ 0)$ and $\emptyset \exists (0 \ 0 \ 1)$ are the

other two basic states. A theorem having this state Σ is sometimes

true, but not for all or for none; but this is not an "incompleteness"

in the extended form EPL (of standard predicate logic), isomorphic

to BA-3, which must have Σ as one of the three basic states,

among the $8 (= 2^3)$ possible states in it. Thus, EPL is as complete

for predicate logic, as SNS is for sentential logic.

A paradox is defined to be an argument which, for both

input states T and F, leads to the output term being contradictory

(i.e. in state X). The essential patterns for this to happen

.iii.

have been analysed and shown to be all derivable from one pattern having the structure of the "Double Statement Paradox, (DSP). This has the statements $\underline{\underline{a}} \sim N = \underline{\underline{b}}$, $\underline{\underline{b}} \sim E = \underline{\underline{a}}$. If the circularity is broken at $\underline{\underline{a}}$, then, starting from $\underline{\underline{a}} = T$, or $\underline{\underline{a}} = F$, we get back only $\underline{\underline{a}} = X$. Vice versa, if the circularity is broken at $\underline{\underline{b}}$, then $\underline{\underline{b}} = X$ always follows from both the inputs $\underline{\underline{b}} = T$ and $\underline{\underline{b}} = F$. However, if the SNS state D is allowed, then $\underline{\underline{a}} = D$ leads back only to $\underline{\underline{a}} = D$ via $\underline{\underline{b}} = D$, and similarly for $\underline{\underline{b}}$. Similarly, $\underline{\underline{a}} = X$, or $\underline{\underline{b}} = X$, as input also leads back to the same state. Hence, the solution of the DSP is that the term $\underline{\underline{a}}$ "may be true or may not be true" (D), or "is impossible"(X). There is no paradox; instead, we can derive the conclusion that "the input statement is invalid" or that "the input statement has the state D". Which of these is true depends on the semantics of the problem.

By combining the two statements in DSP, we obtain the equation $(\underline{\underline{a}} \sim N = \underline{\underline{a}}) \equiv (\underline{\underline{a}} \Rightarrow \neg \underline{\underline{a}}, \neg \underline{\underline{a}} \Rightarrow \underline{\underline{a}}) \equiv (\underline{\underline{a}} \neq \underline{\underline{a}})$.

This is called the Cretan Liar Paradox (CLP) and the best example for \underline{a} is: "What I say is false". Here, both $\underline{a} = T$ and $\underline{a} = F$ lead to $\underline{a} = X$, on breaking the circularity by writing $\underline{\underline{a}} \sim \underline{\underline{a}}$, $\underline{\underline{a}} V \underline{\underline{a}} = \underline{a}$, and we have a paradox. However, for CLP also, both $\underline{a} = D$ and $\underline{a} = X$ lead back to the input state D, or X, as the case may be. Thus, a statement having the pattern of CLP (as "This proposition is not true") may be said to be invalid (X), or may be taken to have the state D; which one is to be chosen will depend on the semantics of the sentence.

By reversing the second statement in DSP, we get the "Contradictory Statements Paradox" (CSP) as follows: $\underline{a} \sim \underline{b}$, $\underline{a} E \underline{b} = \underline{a}$, $\underline{b} V \underline{b} = \underline{b}$. Here also, for $\underline{a} = T$ and $\underline{a} = F$, $\underline{b} = X$, so that it is a paradox in PC; but it is not so, for $\underline{a} = D$ leads to $\underline{b} = D$ (not X) and $\underline{a} = X$ also leads to $\underline{b} = X$. No example of this has been found, but several in the literature belong to a closely related one (CSP-2), which runs as follows: $\underline{a} \sim \underline{b}$, $\underline{a} \sim \underline{b} = \underline{b}$, $\underline{b} V \underline{b} = \underline{b}$. With the added condition $\underline{a} = T$, \underline{b} is X

.v.

and a contradiction follows, as in CSP. Examples of these include Russell's paradox ($\underline{a} \equiv "R$ is the set of all exclusive sets"; $\underline{b} \equiv R \subset R$); and Cantor's paradox ($\underline{a} = "L$ is the set of all sets"; $\underline{b} \equiv "L$ is the largest set"). It is our contention that the CSP-2 is not a paradox, for $\underline{a} = F$ does not lead to a contradiction, and only the assumption $\underline{a} = T$ leads to $\underline{b} = X$, so that we can deduce from it that \underline{a} is F , from the rules of classical propositional calculus. We thus deduce that there is no set R , which is at the same time "exclusive" and the "set of all such sets". Similarly, there is no L which is both a "set, having a power set $P(L)$ larger than itself" and "the set of all sets". In the literature, both these examples are listed as paradoxes having the pattern of CLP, and it is our contention that they are not paradoxes even in classical propositional calculus, but arguments from which valid conclusions can be derived by the reductio ad impossible method.

Thus, paradoxical arguments, which have no solution in FC, have the solution in the states D, or X, of SNS. The choice is X in most cases, saying that the premise of the argument is invalid; but there are situations in which the state D has semantic value, and does not merely state the indefiniteness of the available knowledge. As examples of the relevance of D as the only possible state of a term are given some properties of infinity (which will be extended further in a later report).

Finally, it is shown that, just as SNS (BA-2) and EPL (BA-3) are complete, so is any multivalued logic isomorphic to BA-n. So also, there can be no paradoxes in all logics isomorphic to BA-n, including SNS and EPL. Since PC has necessarily to be extended to SNS, when the paradoxes in it are removed, we thus prove quite generally that there cannot be any incompleteness or paradoxes in any logical system, provided the states in it are extended to be isomorphic to the appropriate Boolean algebra BA-n.

This report in the first draft has been prepared for comment and discussion. Sections 1,2 and 3 are therefore written as outlines, to be suitably modified for publication . Sections 4,5,6 and 7 are fairly complete and contain all the new material. The author would invite critical comments on the contents and presentation of this report.

CONTENTS

	<u>Page No.</u>
1. Introduction	1
2. Boolean algebraic representation of logic	1
3. "Mixed" states and their relevance to the main thesis of this report	3
4. (a) Propositional calculus and SNS logic	5
(b) Absence of closure for PC and extension to SNS	11
(c) Absence of closure for SNS and extension to EPL	15
(d) "Godel's incompleteness theorem does not hold for EPL, but only for QFL	27
5. Aetiology of Paradoxes	30
(a) Double statement paradox (DSP)	32
(b) Contradictory statement paradox (CSP)	36
(c) Cretan liar paradox (CLP)	39
6. Examples of paradoxes and the principles governing their elimination	42
(a) Double statements paradox	42
(b) Contradictory statements paradoxes CSP and CSP-2	45
(c) Cretan liar paradox	53
(d) The state D having semantic value-properties of infinity	55
7. Absence of paradoxes and incompleteness in multi-valued logic	58
(a) Impossibility of incompleteness due to closure of Boolean algebras	58
(b) Non-closure of a new type leading to higher genus Boolean algebras	58
(c) Absence of paradoxes in SNS and EPL and in all logic with n basic states ($n \geq 2$).	60
References	63
List of Tables and Figures	64

Elimination of Paradoxes and Incompleteness from Logic

Introduction:

1./ Logic is the science of reason and therefore the

formulation of the rules in logic cannot be such that they are

sometimes invalid, leading to paradoxes, or incomplete, leading

to possibilities not envisaged in the original rules. It is

our contention that the basic formalism and the set of rules

that are to be applied can be so made as to avoid both of the

above two difficulties — namely existence of paradoxes and

occurrence of incompleteness. This is done by mathematical

representation of logical states and their inter-relationships

in terms of the vector-matrix formalism for Boolean algebra

and logic that we have developed.

Boolean-algebraic representation of logic:

2./ The connection between logic and set theory is a well

established one. So also is the isomorphism between the theory

of sets and Boolean algebra. Thus Boolean algebra BA-n containing

n basic elements, (in which all the states can be represented

by a Boolean vector with n elements (a_1, a_2, \dots, a_n)), is

isomorphous with the algebra of all the subsets of a set

generated by n independent subsets in which no pair of subsets

have a common element. It has been shown in our earlier studies

that BA-2 is the appropriate representation for propositional

calculus (PC) which has two independent states "true"(T) and

"false"(F). Similarly BA-3 has been shown to be the appropriate

representation for quantified predicate logic (QPL) in which the

three independent non-overlapping states are "for all" (\forall),

"for some" (\exists) and "for none" (\emptyset). The formalism that

arises out of this isomorphic representation of logic in terms

of the Boolean algebras BA-2 and BA-3 appears to be very powerful,

and their consequences, particularly in relation to manipulation

of arguments and working out their consequences, have been discussed

in a series of papers [1 to 6].

"Mixed" states and their relevance to the main thesis of this

3./The most interesting consequence of this isomorphic report:

representation by BA-2 and BA-3 is that new "mixed" states

can be obtained by the superposition of the "basic" states in

both BA-2 and BA-3. In the former case, we get two new states

"doubtful" ($D \equiv T \oplus F$) and "contradictory" ($X = T \otimes F$), leading

to four different possible states in the extension of PC. We have

named this the "Syad-Nyaya-System(SNS)" from the Sanskrit words

Syad(= may be) and Nyaya (= logic). In the same way, in BA-3,

in addition to the four well-accepted quantified states

$\forall, \exists, \emptyset, \Delta$ (not for all), four more states are introduced —

namely Σ (for some), Θ (for all or none), Δ (indefinite, for all,

for none, or for some), \emptyset (impossible). This extended predicate

logic (EPL) contains $2^3 (= 8)$ possible states arising from the

mixture of two or more of the three basic states of BA-3 (for

details see [1,2,3]).

The important point to be remembered is that the BA-2 representation of PC automatically leads us, from two to the four

possible states T,F,D,X of SNS, so that the extension from the usual BA-1 algebra of PC to the resulting BA-2 algebra of SNS is inescapable. (See Section 4). So also, the representation of the four common states of QPL, via the three basic states of BA-3, leads necessarily to the eight possible states mentioned above for EPL. Consequently this is the proper set of states which should be taken into account in discussing and working out arguments in predicate logic.

In Section 4, we shall give a very brief outline of the new properties of PC and QPL, in their extended forms, namely SNS and EPL, obtained by using the Boolean algebraic representations BA-2 and BA-3. We shall then show that both sentential logic and predicate logic are complete under the usual operations employed in standard logical arguments. In Section 5, we shall give a general account of paradoxes as they are discussed in standard books and articles on logic, but employing our formalism and nomenclature. We shall show that they can be classified into

three types; and then demonstrate how the paradoxical situations which arise in these examples, (namely that the equations lead to contradiction for both inputs T and F), can be put in the form of normal logical statements which is valid in the extended form of PC, namely SNS, so that the equations are satisfied by either of the new states D or X. In Section 6, examples of paradoxes listed in the literature are examined and the resolution of these are described. In Section 7, we show more generally that there can be no paradoxes in logic and no incompleteness in any particular formulation of multivalued logic. We thus show, in effect, that provided we take into account the right algebraic formulation for logic, then the mathematical condition, that the algebra is complete and does not have contradictions, also leads to the result that the homologous logical formalism is also free of inconsistencies, paradoxes and incompleteness.

4. (a) Propositional calculus and SNS logic: From the example of the conclusion to be derived from the two different types of evidence, namely eye-witness and alibi, regarding the guilt or (see Ref. [1]), absence of guilt of the accused in a murder case / we can show that

there are four possibilities for guilt (\underline{g}): Guilty (\underline{g}_T), not guilty (\underline{g}_F), may be or may not be guilty (\underline{g}_D), and, the two evidences are contradictory (\underline{g}_X). The last/ from the fact that one of the two, namely eye-witness or alibi must be wrong (See Ref [1]). The vector-matrix formulation of this in terms of the two basic states $T = (1 \ 0)$ and $F = (0 \ 1)$ of SNS is obvious, and we obtain $D = (1 \ 0) \oplus (0 \ 1) = (1 \ 1)$ and $X = (1 \ 0) \otimes (0 \ 1) = (0 \ 0)$. $(1 \ 1)$ means that the answer to the question, "can it be true?" is "yes," and also the answer to the question, "can it be false?" is "yes". Similarly $X = (0 \ 0)$ indicates that the answer to the question, "can it be true?" is "no", and the answer to the question, "can it be false?" is also "no". Therefore, there is no logical answer to the conclusion of the statement which has the state $(0 \ 0)$. Consequently, either the premises, or the construction of the argument, is faulty.

Good examples in which the new states D and X arise are the following:

$$\underline{\underline{a}} \sim \underline{\underline{b}}, \underline{\underline{a}} = F \rightarrow \underline{\underline{b}} = "T \text{ or } F" = D \quad (1)$$

Similarly,

$$\underline{a} \wedge \underline{b} = T, \underline{a} = F \rightarrow \underline{b} = X \quad (2)$$

The former one is one of the commonest statements that occur in logic. We have the result that if $\underline{a} \Rightarrow \underline{b}$ is valid, then \underline{b} is necessarily true. However, if \underline{a} is false, we can say nothing about \underline{b} —namely whether it is true or false. Therefore its state in SNS is "T or F" = (1 0) \oplus (0 1) = (1 1).

Using the formalism developed in Refs [1] and [2]

this eqn. takes the form

$$(0 \ 1) \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} = (1 \ 1) \text{ for } \langle a | I | = \langle b | \quad (3)$$

For more details see the appropriate sections of Ref. [1] or [2].

In the same manner, taking Eq.(2) we are given that \underline{a} & \underline{b} is true, which necessarily requires that both \underline{a} and \underline{b} have the logical state T. Therefore, if we have the condition that $\underline{a} = F$ as the second equation in the l.h.s of (2), then there is a contradiction, in that we have $\underline{a} = T$ as the conclusion of the

first statement and $\underline{a} = F$ as the condition of the second

statement, which are mutually contradictory. As mentioned in

Parts I and II, the vidya operator $\underline{\underline{V}}$ has been designed to check

for such contradictions and its algebra is as follows:

$$\underline{a}_1 \underline{\underline{V}} \underline{a}_2 = \underline{a} \rightarrow a_{1\alpha} \otimes a_{2\alpha} = a_\alpha, \quad a_{1\beta} \otimes a_{2\beta} = a_\beta \quad (4)$$

It is readily verified that if $\underline{a}_1 = T = (1 \quad 0)$, $\underline{a}_2 = F = (0 \quad 1)$,

then $a_\alpha = 0$, $a_\beta = 0$ leading to $\langle a \rangle = (0 \quad 0)$.

Since one whole set of paradoxes discussed in the literature have to do with the occurrence of such contradictory states for \underline{a}_1 and \underline{a}_2 , we shall discuss here itself the conditions under which Eq.(4) can lead to identical states for \underline{a}_1 and \underline{a}_2 , when these two, (namely \underline{a}_1 and \underline{a}_2), are negations of each other. The discussion below is based mainly on Boolean algebra and the logical consequences are pointed out thereafter. The negation operator is represented by the matrix $|N|$ (as in 5a) with the properties, as in (5b) and (5c):

$$|N| = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}; \quad (1 \quad 0) \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = (0 \quad 1); \quad (0 \quad 1) \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = (1 \quad 0) \quad (5a, b, c)$$

so that

$$\underline{a}_T \underset{\approx}{\sim} \underline{a}_F , \quad \underline{a}_F \underset{\approx}{\sim} \underline{a}_T \quad (5d,e)$$

For both these basic states T and F for \underline{a} , therefore, we have the result that \underline{a} and $\underline{a} N$ are mutually contradictory, i.e.

$$\underline{a} \underset{\approx}{\sim} (\underline{a} N) = (0 \quad 0) = X \quad (6)$$

However, the Boolean algebra of SNS has, in addition, two other mixed states D and X, and for these we have the following consequences, as in (7) and (8):

$$\underline{a}_D \underset{\approx}{\sim} (1 \quad 1) \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = (1 \quad 1) = \underline{a}_D \quad (7)$$

$$\underline{a}_X \underset{\approx}{\sim} (0 \quad 0) \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = (0 \quad 0) = \underline{a}_X \quad (8)$$

Hence, for the mixed state D, for \underline{a} , Eqn.(6), written in the form (9a, b)

$$\underline{a}_1 = \underline{a}_1 , \quad \underline{a}_2 = \underline{a}_1 N \quad (9a,b)$$

does not lead to X, when $\underline{a}_1 = D$. Instead, we obtain the result in (9c),

$$\underline{a} = \underline{a}_1 = \underline{a}_2 , \quad \text{for } \underline{a}_1 \underset{\approx}{\sim} \underline{a}_2 = \underline{a} , \quad \text{if } \underline{a}_1 = D \quad (9c)$$

The logical consequence of this is very simple. If we are given that a statement \underline{a}_1 leads to a negated form of it as \underline{a}_2 , then \underline{a}_1 and \underline{a}_2 can both be simultaneously valid (true) provided \underline{a}_1 is not one of the two basic states T or F, but is the mixed state D. To put it in words, if something is D, i.e. "it may be true or may be false", then the negation of this has the property that "it may be false or it may be true", which is equivalent to the state D once again. In other words, if the statement is such that it is impossible to assert the absolute truth of it or the absolute falsity of it, then its negation being equivalent to itself does not lead to a contradiction.

It will be noticed that what has been said above has nothing paradoxical about it. We have merely considered the consequences of equating a statement in propositional calculus with its negation and tried to find out if there are any possible logical states for the statement for which this equation does not lead to a contradiction. Then, we find that if $\underline{a} = D$, then, by (7), $\underline{a} \sim N$

is also D so that $\underline{a} = \underline{a} \sim N$.

This property that $\underline{a}_1 = \underline{a}_2$, as in (9a, b) is also true for a_X , i.e. for $\underline{a} = X$, as is shown in (8). However, now

$$\underline{a}_1 = \underline{a}_2 = \underline{a} = X, \text{ where } \underline{a} = \underline{a}_1 \vee \underline{a}_2 \quad (10a, b)$$

The logical consequence of this is that, if something is X, i.e. "it has the property of being impossible" (i.e. $\langle a \rangle = (0 \ 0)$), then its negation $\underline{a}' = \underline{a} \sim N$ is also $(0 \ 0)$ and is impossible. It is a moot point if this can be taken to mean that $\underline{a}' \equiv \underline{a}$, but algebraically (8) shows that $\langle a_X | N \rangle = \langle a_X | \rangle$. We shall show in Section 6 that the states D and X, thus shown to satisfy the equation $\langle a | N \rangle = \langle a | \rangle$, can be made use of for getting over a whole range of paradoxes. Anticipating the discussion in Section 6, we shall call these to be of the Cretan Liar Paradox (CLP) type.

(b) Absence of closure for PC and extension to SNS: Clearly, if we operate only with the two states of BA-1, whose algebra is isomorphic to that of propositional calculus (as discussed in standard books on logic), then the equation $\underline{a} = \underline{a} \sim N$ has no solution

in BA-1, or the Cretan liar paradox is readily a "paradox", and a defect of the logical system of PC. However, we have shown that the inclusion of "reverse" relations in PC lead to the four states T, F, D, X, of SNS, isomorphic to the four elements of BA-2, namely (0 0), (0 1), (1 1) and (0 0).

To be specific, in PC we ask the reverses of relations like —

"a or b is T ; a is T ; What is the state of b?" (11a)

and

"a and b is T, a is F ; What is the state of b?" (11b)

and we get the answers,

$$\underline{b} = D = (1 \ 1) \text{ for (11a)} \quad (12a)$$

$$\underline{b} = X = (0 \ 0) \text{ for (11b)} \quad (12b)$$

This extension of PC to SNS for sentential logic has not been done for the purpose of dealing with paradoxes, but to make the algebra "closed". We find that the inclusion of reverse relations in BA-1 leads to its extension to BA-2 unequivocally, as in (11) and (12). As we will show in Section 6, it also leads to the elimination of the CRP type of paradoxes, including new paradoxes, of the types which we shall formulate.

We shall now ask whether the extended SNS sentential logic is also complete or whether it has to be extended still further, as PC with two states had to be extended to four states by asking

questions as in (11) and (12). The answer to this is that, if only the relations of the type envisaged in standard propositional calculus as discussed in the literature are asked, then SNS and its associated algebra BA-2 are sufficient to represent the logic of all statements in propositional calculus. This is very briefly shown by the following considerations. Thus, there are unary and binary relations in PC, and these are represented by equations of the type,

$$\langle a | z | = \langle b | \quad (13a)$$

and

$$\langle a | z | b \rangle = c_\alpha, \quad \langle a | z^c | b \rangle = c_\beta, \quad \langle c | = (c_\alpha, c_\beta) \quad (13b)$$

When these are reversed, we obtain from (13a) the relation (13c):

$$\langle b | z^t | = \langle a | \quad (13c)$$

in which $|z^t|$ is also one of the 10 (or 16) matrices that are possible for $|z|$, so that no new relations arise from the reversing of unary relations. On reversing binary relations, as shown in

Section 3(b),
Ref [2], we have in effect,

$$\left. \begin{array}{l} \langle a | z | = \langle b | , \text{ if } \langle c | = T \\ \langle a | z^c | = \langle b | , \text{ if } \langle c | = F \\ \langle a | d | = \langle b | , \text{ if } \langle c | = D \\ \langle a | x | = \langle b | , \text{ if } \langle c | = X \end{array} \right\} \quad (14a-d)$$

and once again the operative logical equations are of the unary

type which are contained in (13a). From matrix connectives, applied
of relations,

in the unary and binary form /nothing new comes out by reversing

them. Since all possible logical relations using the standard
(of PC)

connectives "not", "and", "or" / can be represented by 2X2 Boolean

matrices in BA-2, we have the result that such connectives lead

to closure for PC.

This is, however, not the complete picture, for in BA-2
we can also have binary relations involving the connectives

$\underline{\underline{U}}$ and $\underline{\underline{V}}$ and we shall examine below their reversal and show that

this leads to states more complex than those envisaged by BA-2 even
in the extended form of PC, namely SNS.

(c) Absence of closure for SNS and extension to EFL: As

mentioned above, reversal of matrix connectives, namely those

obtained from "not", "and", "or", leads only to relations

expressible once again by 2X2 Boolean matrix connectives in SNS.

However, as mentioned in Ref [1], there are two more connectives,

namely "upon" ($\underline{\underline{U}}$) and "vidya" ($\underline{\underline{V}}$), which lead to binary relations

between SNS logical systems. Their properties have been

extensively dealt with in Refs [1, 2 and 3]. However, this has

been done only ~~at~~^{for} forward operators in the form of relations

of the type $\underline{\underline{a}}_1 \underline{\underline{U}} \underline{\underline{a}}_2 = \underline{\underline{a}}$, $\underline{\underline{a}}_1 \underline{\underline{V}} \underline{\underline{a}}_2 = \underline{\underline{a}}$. If we assume that

there are only two states T and F, then $\underline{\underline{U}}$ and $\underline{\underline{V}}$ have the same

properties as the matrix connectives $\underline{\underline{0}}$ and $\underline{\underline{A}}$ respectively,

and there is no meaning in introducing two new operators $\underline{\underline{U}}$ and $\underline{\underline{V}}$

in the algebra of logic containing only truth values T and F

(as in PC). This fact is well-known. However, the truth table

for $\underline{\underline{U}}$ is quite different from that of $\underline{\underline{0}}$, and similarly that of

$\underline{\underline{V}}$ is quite different from that of $\underline{\sim A}$, in SNS logic. (See Table 4 of [1], and also Eqns (14) and (15) of the same reference.) For ready reference we give below in Table 1 the truth table for the vidya operator $\underline{\underline{V}}$ in the forward direction, namely for the relation,

$$\underline{\underline{a}} \quad \underline{\underline{V}} \quad \underline{\underline{b}} = \underline{\underline{c}} \quad (15)$$

Table 1. SNS truth table for the vidya operator $\underline{\underline{V}}$. (See page 18)

Now if we ask the reverse of (15) — namely, given the SNS states of $\underline{\underline{c}}$ and $\underline{\underline{b}}$, what is the logical state of $\underline{\underline{a}}$? — then we get very strange results. Just as such a question asked for the reversal of the "and" and "or" relations in PC (BA-1) led to states that are not expressible in BA-1, but only in BA-2, here also we find, as the result of the reverse relation

$$\underline{\underline{V}} \quad \underline{\underline{a}} = \underline{\underline{b}} \quad (16)$$

states that are not expressible in BA-2. For this purpose, we have listed below in Table 2 the states of $\underline{\underline{b}}$ corresponding

to the combinations (\underline{c} , \underline{a}), as deduced from the data given in Table 1.

Table 2. Reverse relations obtained from Table 1

We shall indicate the derivation of the various lines in Table 2 as below. Taking Sl.No. 1, if \underline{c} is T and \underline{a} is T, we consider the horizontal row 1 against entry T in Table 1 and look up the corresponding entries for \underline{b} , whenever $\underline{c} = T$.

We thus obtain the two possibilities, T and D. Thus, we get the possibilities that if $\underline{c} = T$ and $\underline{a} = T$ in the relation $\underline{a} \vee \underline{b} = \underline{c}$, then \underline{b} can be T or \underline{b} can be D. However, this is not the complete answer. It should be noted that \underline{b} cannot be F, since the corresponding entry for $(\underline{a}, \underline{b}) = (T, F)$ is X in the first row of Table 1.

Thus, D has to be qualified by saying that it is not F. This is what is written against Sl.No. 1 for \underline{b} in Table 2. The row against Sl.No. 5 follows in exactly the same manner. We have thus the new state "D, but not T", in addition to "D, but not F" found in Sl.No. 1

Table 1. SNS truth table for the vidya operator $\frac{V}{R}$

$$\underline{\underline{a}} \quad \frac{V}{R} \quad \underline{\underline{b}} = \underline{\underline{c}}$$

$\underline{\underline{a}}$	$\underline{\underline{b}}$	T	F	D	X
T		T	X	T	X
F		X	F	F	X
D		T	F	D	X
X		X	X	X	X

Table 2. Reverse relations obtained from Table 1

$$\underline{\underline{c}} \quad \overset{\leftarrow}{\frac{V}{R}} \quad \underline{\underline{a}} = \underline{\underline{b}}$$

Sl. No.	$\underline{\underline{c}}$	$\underline{\underline{a}}$	$\underline{\underline{b}}$
1	T	T	"D, but not F"
2	T	F	Impossible
3	T	D	"Always T"
4	F	T	Impossible
5	F	F	"D, but not T"
6	F	D	"Always F"
7	D	T, F	Impossible
8	D	D	"D, but not T, and not F"

If we take Sl.No. 2 namely $\underline{c} = T$, $\underline{a} = F$, we look at the second row in Table 1 against $\underline{a} = F$ when we find that there is no entry $\underline{c} = T$. Consequently this combination of \underline{c} and \underline{a} is impossible, and cannot lead to any logical state for \underline{b} , except the "impossible" state. (Anticipating later discussion, this "impossible" state is (0 0) in BA-2 and (0 0 0) in BA-3. We have explained the property of this state in BA-3 in Refs [3-6] and utilized it for various purposes.) A similar explanation holds for the entry of "impossible" state in Sl. Nos. 4 and 7.

an Considering Sl.No.3, $\underline{c} = T$, $\underline{a} = D$, the third row shows that there is only one such entry for \underline{c} and this corresponds to $\underline{b} = T$. Hence $\underline{b} = F$ and $\underline{b} = D$ are eliminated, and we indicate it by writing for \underline{b} "Always T" against Sl.No.3. We shall explain below the difference between \underline{b} simple T and always T. A similar explanation holds for "Always F" in row No. 6. We have only the last row 8 to be considered. In this case, we wish to find out

the state of \underline{b} corresponding to $\underline{c} = D$ and $\underline{a} = D$ and looking at row No. 3 in Table 1, we find that $\underline{b} = D$. However, this is not the whole answer. In SNS, D corresponds "T or F", but in the outcome of the combinations of \underline{c} and \underline{a} in row 8, we cannot have the possibilities "T" or "F" for \underline{b} , ^{since} for these lead to T or F for \underline{c} also and not D as ~~was~~ required. Hence it is that the outcome of \underline{a} for \underline{b} is given as "D, but not T, and not F".
 (the combination in Row 8)

We shall now show that the strange new states for \underline{b} outlined in Table 2 are expressible by a Boolean vector-matrix representation in BA-3. We shall not derive the structure of the Boolean algebra which is isomorphic to the states contained in Table 2; instead we shall mention first that such an algebra is BA-3, ~~and~~ state the isomorphism between its three basic states and three out of the various states mentioned in the r.h.s of Table 2, and then show that this Boolean algebra represents all the different states for \underline{b} that is listed in Table 2. Since we already know from the Ref [2-5] that an isomorphism exists between BA-3 and

EPL, we shall use the notation of the latter to denote the corresponding states in Table 2. It turns out that this is indeed most appropriate and the corresponding states, obtained by the reversal of the vidya operator in SNS, and those that are found in the extension of the four states of predicate logic, have a close logical interrelationship.

Thus, the three basic states $(1 \ 0 \ 0)$, $(0 \ 1 \ 0)$, $(0 \ 0 \ 1)$ of BA-3 will represent the following three states from among those listed in Table 2.

$$\text{"Always T"} \mapsto (1 \ 0 \ 0) \equiv \forall \text{ (say)} \quad (17a)$$

$$\begin{aligned} \text{"D, but not T, and not F"} &\mapsto (0 \ 1 \ 0) \equiv \Sigma \\ &(\equiv \text{"Sometimes T"}, \text{ say}) \end{aligned} \quad (17b)$$

$$\text{"Always F"} \mapsto (0 \ 0 \ 1) \equiv \Phi (\equiv \text{"Never T"}, \text{ say}) \quad (17c)$$

In this, the second basic state "D, but not T and not F" has the property that it includes all possibilities where it is sometimes T and sometimes F, but that it excludes the states "Always T", and

"Always F". In this sense, \sum is different from the mixed state $D (= (1 \ 1))$ in SNS logic. Thus, $D \otimes T = T$ and $D \otimes F = F$, although D itself is a mixed state which is sometimes T and sometimes F . However, in the extended BA-3 representation as given in (17) the second state, $(0 \ 1 \ 0)$, also represents a doubtful state, but it excludes the pure states T and F . In consequence $\sum \otimes V = (0 \ 0 \ 0)$ and $\sum \otimes \bar{F} = (0 \ 0 \ 0)$. We mention this here to indicate that the extension from BA-2 to BA-3 produces certain changes in the connotation and representation of the corresponding logical states in SNS and EPL respectively.

(The ideas of basic states and mixed states, as employed in the above Boolean algebraic representation, have quite a lot to do with the properties of pure states and mixed states in the vector space representation of states in quantum mechanics).

Having defined the three basic states of BA-3 by (17a), (17b) and (17c), we can associate three more of the states for b given in Table 2 with their Boolean algebraic counterparts as follows;

"D, but not F" $\mapsto (0 \ 1 \ 0) \oplus (1 \ 0 \ 0) \equiv (1 \ 1 \ 0) \equiv \exists$ (say) (18a)

"D, but not T" $\mapsto (0 \ 1 \ 0) \oplus (0 \ 0 \ 1) \equiv (0 \ 1 \ 1) \equiv \Delta$ (say) (18b)

" Impossible " $\mapsto (0 \ 0 \ 0) \equiv \emptyset$ (say) (18c)

In making these associations, we notice that the D which occurs

in these is the SNS D and includes the possibilities of T and F.

Hence "D but not F" = $\forall \oplus \Sigma = (1 \ 1 \ 0)$ and "D, but not T" =

$\emptyset \oplus \Sigma = (0 \ 1 \ 1)$. Effectively, we erase out the limiting state

T or F which is stated to be not present, and thus get two more EPL

states analogous to well-known states in quantified predicate logic,

as used in standard literature.

On examining (17a-c) and (18a-c), it will be seen that we have obtained the analogues of the four possible states as used in standard first order predicate logic — "for all" (\forall), "there exists" (\exists), "for none" (\emptyset) and "not for all" (Δ). Hence

we can say that effectively we have generated predicate logic and

its four logical states from the reverse relations obtained using

the vidya operator and using different states for \underline{c} and \underline{a} in this

relation $\subseteq \overset{\leftarrow}{\sum} a = b$. Incidentally, there are two more states that are obtained by checking through the list of new states obtained from the above reverse relation, namely $\sum = (0 \ 1 \ 0)$ which corresponds to EPL state "some", and $\emptyset = (0 \ 0 \ 0)$ corresponds to EPL state "impossible". These two states are not considered in standard literature on predicate logic. In fact, as will be seen in the Sections 5 and 6, the state \sum , excluding \forall and \emptyset , is considered to be an indication of the incompleteness of first order predicate logic. ~~Obviously it is an extra state obtainable by manipulation with the four QPL states, for example,~~

$$\exists \otimes \Lambda = (1 \ 1 \ 0) \otimes (0 \ 1 \ 1) = (0 \ 1 \ 0) \quad (19)$$

We have indicated this in Ref [3] (p. 340) and it is certainly true that this state is outside the range of states acceptable for logical terms in standard predicate logic. However, as mentioned earlier, the correct way of looking at these states is by their isomorphism with the corresponding states of BA-3 and

we find that BA-3 has this state $\sum = (0 \ 1 \ 0)$, as one of the basic states of the algebra. Hence, there is no incompleteness if the proper extended set of states of predicate logic are taken into account. In that sense, we shall complete the set of six states for Table 2 - in (17) and (18) and obtain two more so as to complete the set of eight states which the BA-3 representation associated with it ^{has} ~~as~~ in its list. Thus we define,

$$\text{"D, or T, or F"} = (0 \ 1 \ 0) \oplus (1 \ 0 \ 0) \oplus (0 \ 0 \ 1) = (1 \ 1 \ 1) \equiv \Delta \text{(say)} \quad (20a)$$

and

$$\text{"T or F, but not D"} = \text{"Always T" or "Always F"} = (1 \ 0 \ 0) \oplus (0 \ 0 \ 1) = (1 \ 0 \ 1) \equiv \Theta \text{(say)} \quad (20b)$$

This way of arriving at the eight states of BA-3 and their logical correspondences is very instructive, in that we are able to see the full scope of the meaning of these eight states in predicate logic. The really new state $\sum = (0 \ 1 \ 0)$ comes out as "D, but not T and not F", a state which comes only in a complicated way from the four states of classical predicate logic as in Eq.(19).

We are able to see straightaway the existence of three primitive

states in the logic and a re-examination of the equations in

(17) to (20) will show that they have a coherent structure and

form a complete set of $2^3 = 8$ states that correspond to BA-3,

which, therefore, is the right set to be adopted for dealing with

predicate logic. Thus, SNS (BA-2) on extension leads to EPL

(BA-3), just as PC(BA-1) on extension leads to SNS(BA-2). Also,

SNS is complete (closed) with respect to matrix operators used

in unary forward, unary reverse, binary forward and binary reverse

relations. It is also closed with respect to Boolean operators

U and V applied in binary forward relations; but it has to be

extended to a Boolean algebra of higher genus when these are

reversed. Therefore, if we say that sentential logic (which

requires the SNS form of algebra for its full representation)

is closed, we mean only what has been stated above — i.e. it

is closed with respect to matrix operators of all types and only

for Boolean operators applied in the forward direction. Let us consider the corresponding closure properties of EPL which will be done in the next subsection 4(d).

(d) "Godel's incompleteness theorem does not hold for EPL, but only for QPL: We have seen above that SNS is closed in the classical sense i.e. with respect to matrix operators as mentioned there. It is very easy to see that EPL (BA-3) is also closed to the same extent, i.e. for matrix and Boolean operators applied in exactly the same way (see the general theory for matrix connectives in the theory of relations in Ref.[2] sections 2 & 3).

What we see from that general theory is that the eight states of EPL form a closed set when matrix connectives are applied to them for unary forward, unary reverse, binary forward and binary reverse relations. Also this set is closed with respect to binary forward Boolean operators ∇ and ∇' as in $a \oplus b = c$ and $a \otimes b = c$.

If a and b are one of the eight states of EPL, then c is also one of these. Hence predicate logic, in its extended form EPL, is

Logic

as much a closed system, as sentential logic is considered to be so.

(We are not considering the reversal of Boolean operators as ^{in EPL,}

it is too complicated; but the existence of this possibility is to be remembered while considering completeness of the extended predicate logic).

As mentioned in the headings of this subsection, the completeness of EPL does not mean that the smaller set of four states $\forall, \exists, \Phi, \Lambda$ form a closed set. In fact, by applying the Boolean operators of BA-3, namely "upon" (\sqcup) and "vidya" (\sqvee), we obtain states other than these four as in (21a-d).

$$\forall \sqcup \Phi \equiv (1 \ 0 \ 0) \oplus (0 \ 0 \ 1) = (1 \ 0 \ 1) = \Theta \quad (21a)$$

$$\forall \sqvee \Phi \equiv (1 \ 0 \ 0) \otimes (0 \ 0 \ 1) = (0 \ 0 \ 0) = \emptyset \quad (21b)$$

$$\exists \sqcup \Lambda \equiv (1 \ 1 \ 0) \oplus (0 \ 1 \ 1) = (1 \ 1 \ 1) = \Delta \quad (21c)$$

$$\exists \sqvee \Lambda \equiv (1 \ 1 \ 0) \otimes (0 \ 1 \ 1) = (0 \ 1 \ 0) = \Sigma \quad (21d)$$

Hence if suitable sentences are written in QPL using only the four standard states of predicate logic as in the literature,

but using the properties of \cup and \vee , then the resultant conclusion may have the BA-3 states corresponding to one of the four additional states given in (21a-d). In particular, it should be mentioned that the combined information from two such QPL statements in the QPL states \exists and Λ can lead via the vidya operator \vee to the state Σ . We have seen that Σ corresponds to the property of α statement \forall $A(x)$ being "true for some x , but not always true or never true". This statement is the famous "Godel's incompleteness theorem, which we $\frac{1}{\text{have}}$ proved to be possible, in one step in Eq (21d). What we say in effect is that if we have only the QPL states available, then by using the logic of the vidya operator as $\frac{1}{\text{the}}$ connective, we can get out of them the Godel's state Σ . At the same time, we also say that the occurrence of Σ is not an indication of incompleteness of predicate logic, for this state Σ occurs in the extended form EPL of QPL, and, together with seven other states, forms a closed system for predicate logic. This closed system has the Boolean

algebraic structure of BA-3 and it should be noted that some of the states of BA-3 are absolutely necessary for properly representing the properties of QPL. Hence the extension to EPL is not artificially made, but is a natural consequence of obtaining the whole BA-3 set of states to represent the properties of predicate logic. For ready reference, we may mention that additional these four states are Θ (\equiv All or none), \emptyset (\equiv impossible), Δ (\equiv universal doubt), Σ (\equiv some, but not all and not none).

We shall give examples of the theory mentioned in this subsection when dealing with paradoxes and incompleteness in the next sections 5, 6 and 7.

5. Aetiology of Paradoxes

In this section, we shall first consider in general what are the broad properties of argument that lead to a paradox.

Most generally they / may be stated as follows. We carry out the steps listed for the argument and finally arrive at two truth

values \underline{a}_1 and \underline{a}_2 for the conclusion \underline{a} of the argument, which, however, have the property that

$$\underline{a}_1 \underset{\sim}{\vee} \underline{a}_2 = X = \text{"impossible"} \quad (22)$$

Also, the steps of the argument, and the truth values of the terms that are input into the argument are all either unquestionable, or, for all possible states of the inputs, the result (22) follows. Then, the argument is said to be a paradox. The result in Eq.(22) can occur, if there occur the equations:

$$\underline{a}_1 = \underline{s}, \quad \underline{a}_2 = \neg \underline{s} = \underline{s} \underset{\sim}{\wedge} N \quad (23)$$

and further \underline{a}_1 and \underline{a}_2 are restricted to the possible truth values T and F. Put in words, a paradoxical statement always leads to the conclusion that something is true and that it is false at the same time. Then if true and false are the only states that are permissible (as in propositional calculus), then $\underline{a}_1 \underset{\sim}{\vee} \underline{a}_2$ will lead to the impossible state $X = (0 \quad 0)$. (The extension of these criteria for a paradoxical statement in predicate calculus is discussed in Section 7.)

Depending upon how a₁ and a₂ in (22) are arrived at, there is a whole range of possibilities of writing paradoxical arguments. We shall, however, consider, in detail, only three structural types that cover practically all examples — namely Double Statement Paradox (DSP), Cretan Liar Paradox (CLP) and Contradictory Statement Paradox (CSP).

We shall give here the essence of the argument, with a symbolic presentation of the essence of the ideas. Actual examples are given in the next section 6.

(a) Double statement paradox (DSP): The essential structure of

this type of argument can be given by the two equations

$$\underline{\underline{a}} \underset{\sim}{N} = \underline{\underline{b}} \quad ; \quad \underline{\underline{b}} \underset{\sim}{E} = \underline{\underline{a}} \quad (24a,b)$$

If we call the input to (24a) as a₁ and the output of (24b) as a₂, then it is readily seen that

$$\text{For both } \underline{\underline{a}}_1 = T \text{ and } \underline{\underline{a}}_1 = F, \quad \underline{\underline{a}}_1 \underset{\sim}{V} \underline{\underline{a}}_2 = X \quad (24c)$$

The graph of this process, which leads to a contradiction,

irrespective of whether we start with $\underline{a} = T$, or $\underline{a} = F$, is shown in Fig 1(a and b).

Fig.1. Graph of the essential logic of the Double Statement Paradox

- (a) By breaking the circularity at \underline{a}
- (b) By breaking the circularity at \underline{b}

In Fig.1(a), we have broken the circularity of the argument

in (24) at the term \underline{a} (following the procedure mentioned in Part \bar{V} , Section 4(c)), and obtained $\underline{\underline{a}}_1$ and $\underline{\underline{a}}_2$, which are to be compared by the vidya operator to give the resultant \underline{a} . Then the results given below in (25) follow.

$$\underline{\underline{a}}_1 = T \rightarrow \underline{b} = F \rightarrow \underline{\underline{a}}_2 = F \rightarrow \underline{a} = X \quad (25a)$$

$$\underline{\underline{a}}_1 = F \rightarrow \underline{b} = T \rightarrow \underline{\underline{a}}_2 = F \rightarrow \underline{a} = X \quad (25b)$$

As mentioned above, both the states T and F introduced for $\underline{\underline{a}}_1$

lead to the contradictory state as output for \underline{a} , showing that the argument is invalid, both for the inputs T and F. Hence it

.34.

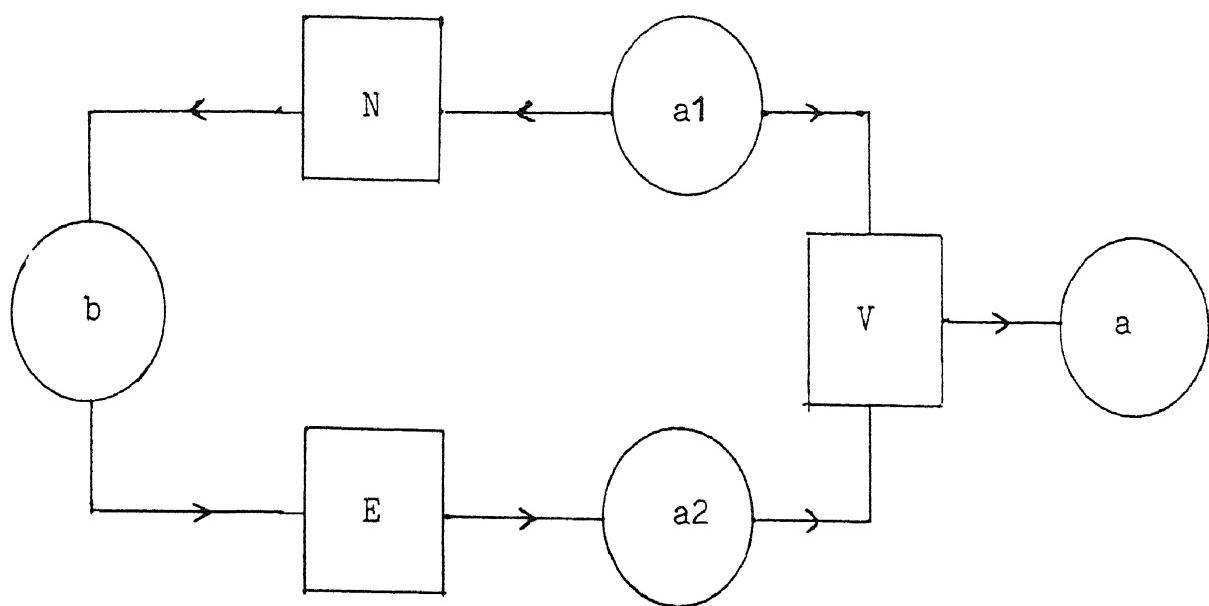


Fig. 1(a)

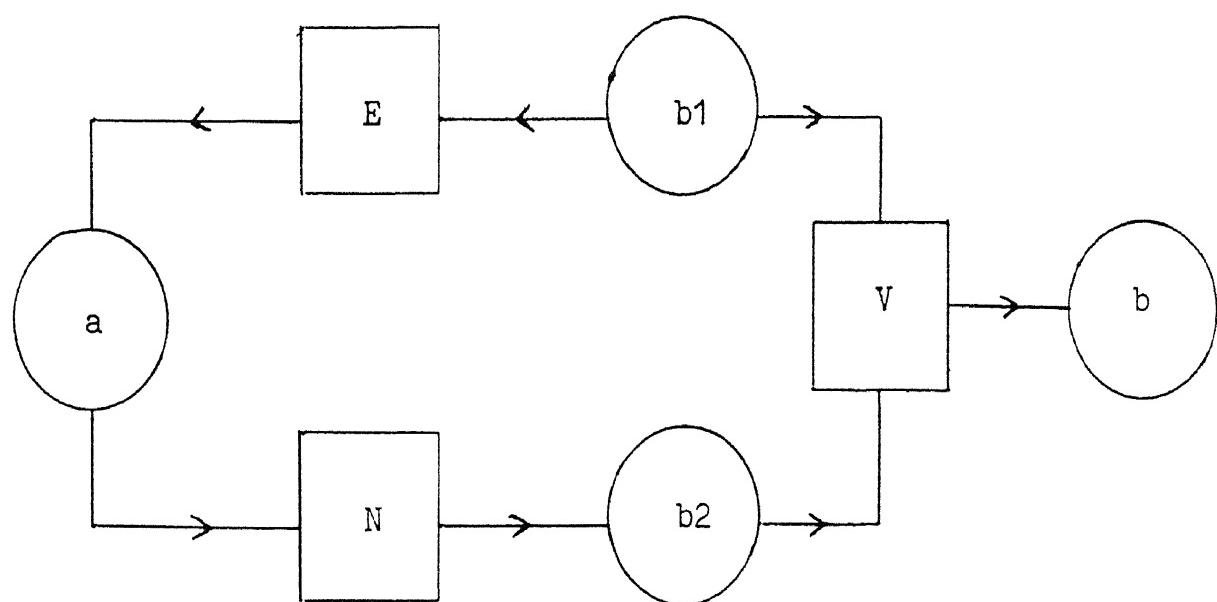


Fig. 1(b)

Fig.1. Graph of the essential logic of the Double Statement Paradox

- (a) By breaking the circularity at a
- (b) By breaking the circularity at b

forms a paradox, since no state in propositional calculus for the input leads to a valid result.

We may also break the circularity of the argument in (24) at the term $\underline{\underline{b}}$ and obtain Fig. 1(b). For this, we obtain, analogous to (25a,b), the results

$$\underline{\underline{b}}_1 = T \rightarrow \underline{\underline{a}} = T \rightarrow \underline{\underline{b}}_2 = F \rightarrow \underline{\underline{b}} = X \quad (26a)$$

$$\underline{\underline{b}}_1 = F \rightarrow \underline{\underline{a}} = F \rightarrow \underline{\underline{b}}_2 = T \rightarrow \underline{\underline{b}} = X \quad (26b)$$

Thus, we see that, even if we start with $\underline{\underline{b}}$, the output of the argument is X , for both the states T and F of $\underline{\underline{b}}$, so that it is a paradox.

Eqs. (24a,b) are typical of all circular arguments that are paradoxes. We have a path from $\underline{\underline{a}}$ to $\underline{\underline{b}}$ and another back from $\underline{\underline{b}}$ to $\underline{\underline{a}}$, and, effectively, one has the unary connective \mathcal{N} connecting $\underline{\underline{a}}$ with $\underline{\underline{b}}$, and the other has the unary connective \mathcal{E} . (See below for an even shorter paradoxical statement obtained by combining the two equations (24a) and (24b). This yields the CLP).

In fact, the DSP is the most crucial paradoxical argument to be considered, for the other two, namely CSP and CLP, can both be worked out by suitably manipulating the two equations in it. We shall give a very simple example of DSP in Section 6 below.

It is obvious that the state $\underline{\underline{a}}_1 = D$ leads back to $\underline{\underline{a}}_2 = D$, so that $\underline{\underline{a}}_1 \vee \underline{\underline{a}}_2 = \underline{\underline{a}}$ gives the same state as the input state $\underline{\underline{a}}_1$. (This is in SNS). This also happens for $\underline{\underline{a}}_1 = X$. Hence the solution to the double statement paradox, (as stated in (24)) is that the entities denoted by the input $\underline{\underline{a}}_1$ can have the possibilities,

"It may be true or not true"; or "It does not exist". when the output $\underline{\underline{a}}_2$ is the same as the input $\underline{\underline{a}}_1$, and the paradox is eliminated.

(b) Contradictory statements paradox (CSP): The statements in (24a,b,c) can also be written equivalently as in (26a,b,c), in which we

obtain two outputs $\underline{\underline{b_1}}$ and $\underline{\underline{b_2}}$, starting from \underline{a} as input (whose graph is shown in Fig. 2).

$$\underline{a} \underset{\sim}{\sim} \underline{\underline{b_1}} ; \quad \underline{a} \underset{\sim}{\sim} \underline{\underline{b_2}} ; \quad \underline{\underline{b_1}} \vee \underline{\underline{b_2}} = \underline{\underline{b}} \quad (26a,b,c)$$

On inputting $a = T$ and F , we obtain,

$$\underline{a} = T \mapsto \underline{\underline{b_1}} = F ; \quad \underline{\underline{b_2}} = T, \text{ so that } \underline{\underline{b}} = X \quad (27a)$$

$$\text{and } \underline{a} = F \mapsto \underline{\underline{b_1}} = T ; \quad \underline{\underline{b_2}} = F, \text{ so that } \underline{\underline{b}} = X \quad (27b)$$

Fig. 2. Logical graph of the contradictory statements in (a) the CSP paradox as given in Eqs (26a,b,c); (b) the CSP-2 paradox in (30a,b,c,d).

Hence, $\underline{\underline{b}}$ becomes "impossible" (X) for both the inputs T and F for \underline{a} , so that the argument symbolized in (26) (which is equivalent to DSP), is again a paradox. We shall call this as the Contradictory Statements Paradox (CSP), since the two statements in (26a) and (26b) are mutually contradictory. It is generic of all examples where, starting from an input \underline{a} , we follow two paths, and arrive at mutually contradictory results $\underline{\underline{b_1}}$ and $\underline{\underline{b_2}}$ for $\underline{\underline{b}}$, and this is so, irrespective of whether the input \underline{a} has the state T or F . We shall discuss this further also in Section 6.

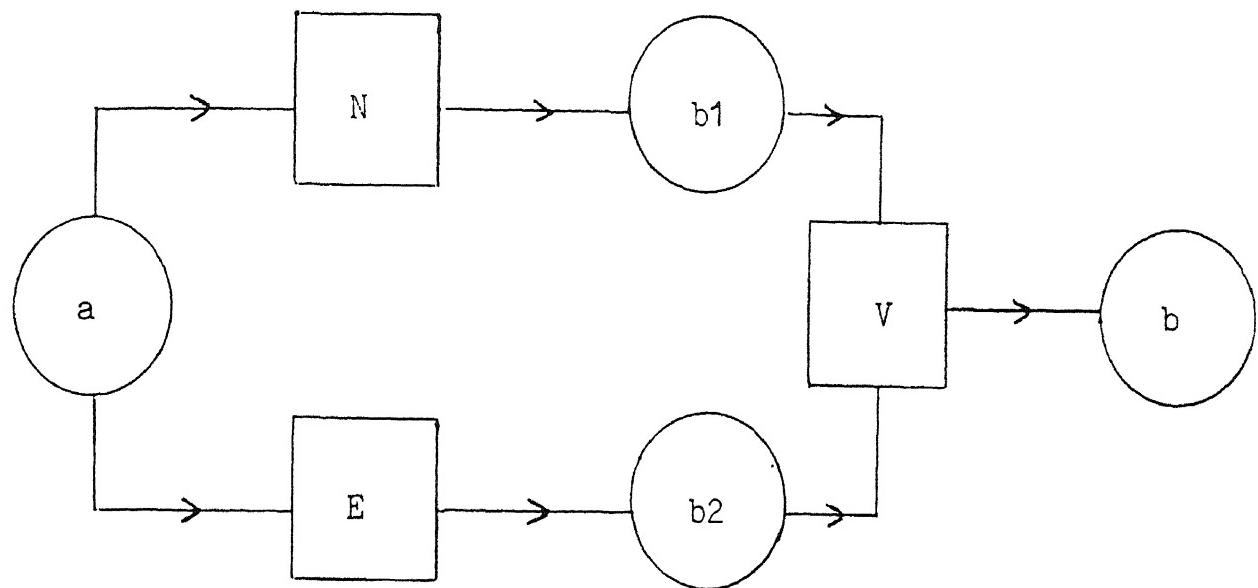


Fig.2(a)

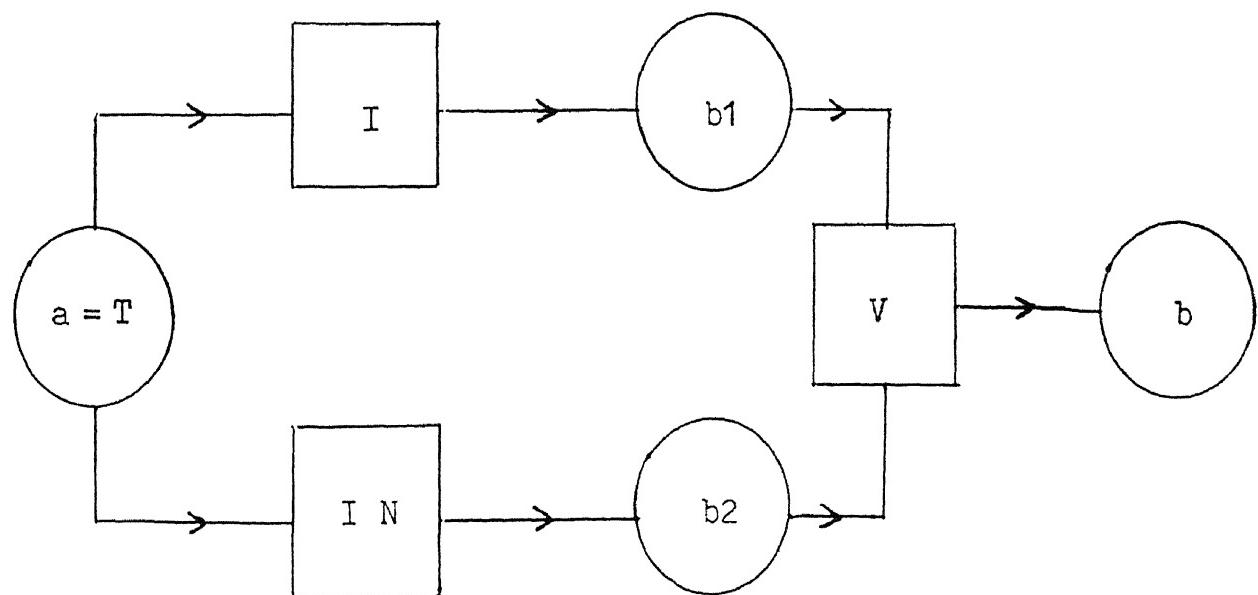


Fig.2(b)

Fig.2. Logical graph of the contradictory statement in
 (a) the CSP paradox as given in Eqs. (26a,b,c);
 (b) the CSP-2, as in (30a,b,c,d).

(c) Cretan liar paradox(CL): Another way of manipulating equations

(24a) and (24b) of DSP is to combine the two sequential unary

statements in it to obtain a single statement, as in (27) below:

$$(\underline{\underline{a}} \text{ } N = \underline{b}, \underline{b} \text{ } E = \underline{\underline{a}}) \equiv (\underline{\underline{a}} \text{ } N \text{ } E = \underline{\underline{a}}) \equiv (\underline{\underline{a}} \text{ } N = \underline{\underline{a}}) \quad (27)$$

The graph of this has the structure shown in Fig. 3(a), and

breaking the circuit at $\underline{\underline{a}}$, we obtain the graph shown in Fig.3(b).

In this, the $\underline{\underline{a}}$ on the l.h.s of (27) is designated as $\underline{\underline{a}}_1$ and that on the r.h.s. as $\underline{\underline{a}}_2$, and we check their mutual compatibility by the vidya operator, to give $\underline{\underline{a}}_1 \underset{\approx}{\sim} \underline{\underline{a}}_2 = \underline{\underline{a}}$.

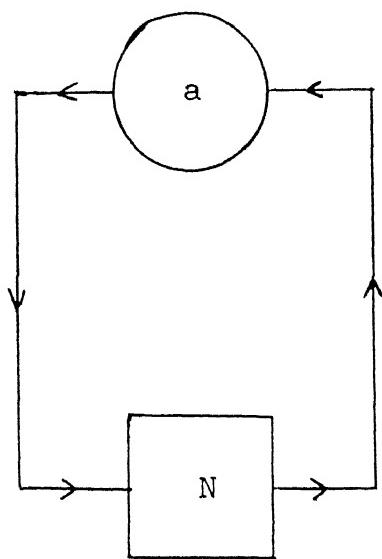
Fig.3 Logical graph of the Cretan Liar Paradox;

- (a) Circular argument as in Eqn. (27)
- (b) With circularity broken and introducing the vidya operator.

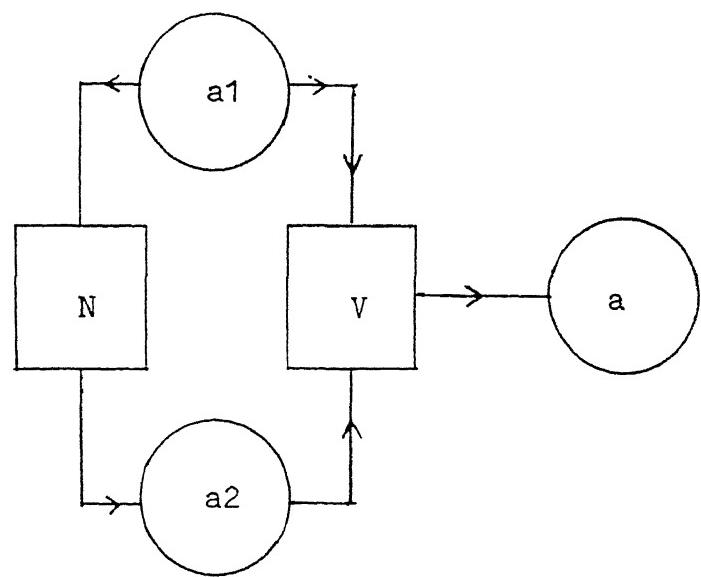
It is readily verified that the argument as formulated in

Fig.3(b) give the results

$$\underline{\underline{a}}_1 = T \rightarrow \underline{\underline{a}}_2 = F \rightarrow \underline{\underline{a}} = X ; \quad \underline{\underline{a}}_1 = F \rightarrow \underline{\underline{a}}_2 = T \rightarrow \underline{\underline{a}} = X \quad (28a,b)$$



(a)



(b)

Fig.3 Logical graph of the Cretan Liar Paradox;

- (a) Circular argument as in Eqn. (27).
- (b) With circularity broken and introducing the vidya operator.

Thus, \underline{a} can be neither T nor F, if $\underline{a} \sim \underline{a}$ of Eq. (26) is to be satisfied. It is given the name "Cretan liar" paradox^(CLP), since this equation covers the logic of the well-known paradox in Greek logic with this name, in which the Cretan says, in effect, "What I say is not true". Is that statement itself true or false? Here \underline{a} stands for the statement, and the contents of the statement give the equation $\underline{a} \sim \underline{a}$. Then, if we start with $\underline{a} = T$, an endless succession of truth values $\underline{a} = F$, $\underline{a} = T$, $\underline{a} = F$, ..., is obtained, by successively applying the equation, so that \underline{a} can be "neither T nor F", if it is to be only one of T or F (which is a paradox in PC). As already mentioned earlier, $\underline{a} = D$ and $\underline{a} = X$ are possible solutions of $\underline{a} \sim \underline{a}$ in SNS, and we shall indicate their reasonableness and relevance to the CLP in Section 6.

6. Examples of paradoxes and the principles governing / elimination

In this section, we shall take the three typical forms of paradoxical arguments given in the last section in a symbolical language and give examples from practical logic for each of them. While doing this, we shall also indicate how the resolution of the paradoxes as mentioned earlier, namely by taking into account the SNS states D and X, is carried out. It is believed that these examples will make clear how other examples of similar paradoxes can be manipulated so as to eliminate the paradoxical nature of the arguments in them.

(a) Double statements paradox: The best example of illustrating this, in the form of Eqs (24a, b), is as follows:

The next statement is false (29a)

The previous statement is true (29b)

If we take \underline{a} to indicate the truth value of the first statement and \underline{b} to indicate the truth value of the second statement, then

it is obvious that the two are related by the Eqns (24a,b).

Then, neither of the states T or F for \underline{a} or \underline{b} , leads back to itself. Instead, in all cases, it leads back to the opposite state, and Eqn.(24c), namely $\underline{\underline{a}_1} \vee \underline{\underline{a}_2} = X$, is the result.

Thus, in both cases, the argument is self-contradictory and is therefore invalid irrespective of whether \underline{a} is T or F, or \underline{b} is T or F.

Since, in propositional calculus, there are no other input states available, we get the result that the argument composed of the two sentences (29a) and (29b) is a paradox and does not have any solution.

Does this mean that the argument is necessarily "self-contradictory", because this is the first conclusion that one would arrive at from the conditions $\underline{\underline{a}_1} \vee \underline{\underline{a}_2} = X$ and $\underline{\underline{b}_1} \vee \underline{\underline{b}_2} = X$. It is not necessarily so, because we can show that the state D as input for \underline{a} leads to D for \underline{b} , and back again to D for \underline{a} , leading to no contradiction. Hence a non self-contradictory solution to the two Eqns (29a and b) is the state D for \underline{a} and \underline{b} , which means

of the
that neither statements can be assumed to be either true at
all times or false at all times. Its truth or falsity cannot be
asserted, but the best that we can do is to say that it "may be
true" or "may be false".

This solution appears to be trivial and that it has no
information content about a or b, because when we know nothing
about a statement a, we say that it is in the state D. For
example, we can readily show that the equations
 $\underline{c} \circ \underline{d} = T$ and $\underline{c} = T$ lead only to $\underline{d} = D$, in which case
we can say nothing definite about the state \underline{c} . However, we shall
show in Section 6(d) that, under certain conditions, the state D
for the truth value of a logical statement can have a profoundly
important value. Since the appreciation of this will need a
better acquaintance with our methods of dealing with paradoxes,
we shall discuss this only in Section 6(d) after we have considered
examples of all the three types of paradoxes mentioned in ^{the} Section 5.

CSP and CSP-2

(b) Contradictory statements paradoxes / The paradox CSP, whose

symbolic representation is given in Eqns (26a,b,c) starts from

the same statement a and arrives at two logical states b1 and b2

for the statement b via two different paths and it is found

that b1 and b2 are self-contradictory, irrespective of whether

a is true or a is false. In order to make the discussion

sufficiently general and to consider the examples available in

the literature of mathematics, particularly in set theory, we

shall slightly modify this and give it another form. This can

be written symbolically as CSP-2, below:

$$\underline{\underline{a}} \text{ I } = \underline{b1}, \quad \underline{\underline{a}} \text{ I } \text{ N } = \underline{\underline{b2}}, \quad \underline{\underline{a}} = T; \quad \underline{\underline{b}} = \underline{b1} \vee \underline{\underline{b2}} = X \quad (30a, b, c, d)$$

In this, the implications I and I N are used, rather than the

equivalence operators E and N. However, the starting point a

is given to be true without question and in that case we get a

self-contradiction. It may be questioned as to how two mutually

contradictory results b1 and b2 can be arrived at from the same

assumption — namely the truth of a. Before giving examples,

we shall indicate that this ^{only} happens because the statement a

contains two conditions in it, one of which leads to b₁ and

the other to b₂, and it is quite possible that these two

consequences b₁ and b₂ are mutually contradictory. Eqn.(30)

only gives the logical structure of this type of paradox.

The genesis of the two mutually contradictory conclusions is a

matter of semantics and as will be seen from the three

examples in basic mathematics which are given below, such a

situation is quite possible. // We shall first take the Russell's

paradox, which may be stated as follows:

"An exclusive set is one that does not contain
itself as one of its numbers. If \mathcal{R} is the
set of all exclusive sets, does it contain
itself or not?".

Thus we have two statements,

a \equiv " \mathcal{R} is the set of all exclusive sets" (32a)

b \equiv $\mathcal{R} \subset \mathcal{R}$ (32b)

The paradox arises by assuming the two properties namely,

" \mathcal{R} is a set of all sets" and " \mathcal{R} is an exclusive set" $\not\rightarrow$ true

The first leads to the condition that $\mathcal{R} \subset \mathcal{R}$ so that $\underline{a} \Rightarrow \underline{b}$

and $\underline{b}_1 = T$. The second one, however, leads to the result that

$\mathcal{R} \not\subset \mathcal{R}$, or $\underline{a} \Rightarrow \neg \underline{b}$ and $\underline{b}_2 = F$. Hence we have the result

(30d) namely $\underline{b} = \underline{b}_1 \vee \underline{b}_2 = X$, or a contradiction.

Thus, the assumption that \underline{a} is true, namely that there is a set \mathcal{R} which consists of all exclusive sets, leads to the conclusion that \mathcal{R} is both contained in \mathcal{R} and not contained in \mathcal{R} .

This is Russell's paradox.

to the question in (31)

The answer is that the assumption made at the start, namely that $\underline{a} = T$, is invalid, and that \underline{a} should be F ; or that "there exists ^{an} \mathcal{R} which is the set of all exclusive sets" is an invalid statement.

the paradox, namely

In fact, such a resolution of that the obviously permissible statement which we start with is an impossible one, occurs in an

even simpler example, namely the Burali-Forti Paradox: This can be stated as follows:

"If \mathcal{N} is the set of all integers from 1 to ∞ is its ordinal number N the largest integer?" (33)

In this, the statements \underline{a} and \underline{b} are

$\underline{a} \equiv$ "There exists N , which is the ordinal number of \mathcal{N} , which is the set of all integers from 1 to ∞ ." (34a)

$\underline{b} \equiv$ " N is the largest integer." (34b)

Here also there are two conditions \underline{a} , which lead to mutually contradictory consequences for \underline{b} . Thus, \underline{a} that N is the ordinal number of \mathcal{N} yields the result that N is larger than every ordinal number, or $\underline{b} = T$. On the other hand, the condition $\underline{b} = F$ that N is an integer demands that there is another integer greater than itself, since every integer has a successor; this means that $\underline{b} = F$. Hence we obtain the contradiction in the style of CSP-2.

The solution, as in the case of Russell's paradox is that the statement (33) is itself invalid; i.e. we cannot talk of the

"ordinal number of the set of all integers". As is well-known, in Peano's axiomatic formulation of the theory of numbers, it is stated that every integer (natural number) has a successor and that there is a first integer; but no mention is made about the last integer. The last statement is quite correct, because the assumption of the existence of a last integer N for the set of integers 1 to ∞ leads to the contradiction mentioned above.

The third example — namely Cantor's paradox — is even shorter. It goes as follows:

"If \mathcal{L} is the set of all sets, is it the largest set?" (35)

Here, we use the fact that, from every set \mathcal{S} , we can construct its power set $P(\mathcal{S})$ (i.e. the set of all subsets of \mathcal{S}), which is larger than itself. Hence, $P(\mathcal{L})$ is a set larger than \mathcal{L} , and the answer to (35) is "No". On the other hand, since \mathcal{L} is said to contain all sets, answer to (35) is "yes". The two are mutually contradictory.

In our notation, the statements \underline{a} and \underline{b} are

$$\underline{a} \equiv "L is the set of all sets" \quad (36a)$$

$$\underline{b} \equiv "L is the largest set" \quad (36b)$$

and we obtain that $\underline{a} \Rightarrow \underline{b}$ and $\underline{a} \Rightarrow \neg \underline{b}$ from the two properties "all" and "set" of L . Consequently, if \underline{a} is true, there is a contradiction, leading to the conclusion that \underline{a} is false, or that "there is no set of all sets".

Thus, we have seen three examples of CSP-2. The author has not been able to trace any example of CSP as such, where, for both $\underline{a} = T$ and for $\underline{a} = F$, the conclusions are of the form $\underline{a} \Rightarrow \underline{b}$ and $\underline{a} \Rightarrow \neg \underline{b}$, which is what is demanded for $\underline{a} \equiv \underline{b}$ and $\underline{a} \neq \underline{b}$ at the same time. This is surprising, for CSP is equivalent in its logic to DSP, for which there are examples. However, we know that some theorems of the form $\underline{a} \Rightarrow \underline{b}$ can be proved only in the form $\neg \underline{b} \Rightarrow \neg \underline{a}$ in a reductio ad impossible proof, but it may not be possible to formulate the direct proof.

It should be noted that, in all the three examples given in this section, $\underline{a} = T \rightarrow \underline{b} = X$, and we can then deduce that $\underline{a} = F$ even using BA-1 theory of propositional calculus, without any recourse to SNS states. There is no "paradox" at all in all these examples, for $\underline{a} = F$ does not lead to a contradiction. The contradiction, in fact, arises from two different assumptions in \underline{a} , which we may call \underline{p} and \underline{q} , which lead to

$$\underline{p} \Rightarrow \underline{b}_1, \quad \underline{q} \Rightarrow \underline{b}_2 = \neg \underline{b}_1, \quad \underline{b}_1 \underset{\sim}{\vee} \underline{b}_2 = X \quad (37a,b,c)$$

so that, $\underline{p} = T$ and $\underline{q} = T$ leads to a contradiction X. Hence, we can deduce that

$$\underline{p} \not\equiv \underline{q} = F \quad (38)$$

i.e. $\underline{p} = T$ demands $\underline{q} = T$ and $\underline{q} = T$ demands $\underline{p} = F$.

For example, in Russell's paradox, there is no "exclusive set", which is at the same time the "set of all such sets". One or other is not true.

However, in the literature, these are shown as paradoxes of CLP type, viz $\underline{a} \equiv \neg \underline{a}$ (see [7], p 128 for example). The argument for Russell's paradox is given in the form

$$(R \subset R) \Rightarrow (R \not\subset R) \text{ and } (R \not\subset R) \Rightarrow (R \subset R) \quad (39a,b)$$

so that

$$(R \subset R) \equiv (R \not\subset R), \text{ a contradiction} \quad (39c)$$

Denoting $(R \subset R)$ by \underline{b} , as in (32b), it is to be noted, ^{however,} that

it is not the \subset in $R \subset R$ that leads to $\underline{b} \Rightarrow \neg \underline{b}$ as in (39a), but it is the "exclusive" in \underline{a} of (32a) that leads to $\underline{a} \Rightarrow \neg \underline{b}$. Similarly, it is not the $\not\subset$ in $R \not\subset R$ that leads to $\neg \underline{b} \Rightarrow \underline{b}$ as in (39b), but it is the "all" in \underline{a} of (32a)

that leads to $\underline{a} \Rightarrow \underline{b}$. Thus, Russell's paradox, which is clearly a CLP paradox, if it is stated in the form $\underline{b} \Rightarrow \neg \underline{b}$

and $\neg \underline{b} \Rightarrow \underline{b}$, is not a paradox at all when it is formulated

in the form of the CSP-2 equations, but only a contradiction,

leading back, from $\underline{a} = T$ and $\underline{b} = X$, to $\underline{a} = F$, a perfectly admissible deduction in PC.

The situation is similar for the Burali-Forti paradox and Cantor's paradox. In fact, quite generally, CSP-2 is not a paradox, but only indicates an inner contradiction in the input statement \underline{a} , of the form $\underline{p} \& \underline{q}$. If $\underline{b} = X$, then it follows straightaway that either \underline{p} , or \underline{q} , is F; both properties \underline{p} and \underline{q} cannot be present in the input statement. The author believes that this point of view must be carefully considered.

(c) Cretan liar paradox: An example of this has already been

given in Section 5(c) of the form "what I say is not true".

If this stands for \underline{a} , then $\underline{a} \Rightarrow \neg \underline{a}$ leading to the

paradox. Any number of examples can be set up which has this

form, or the equivalent form with two sentences in the double

statement paradox. In these cases, the question is as to what

is the way of resolving the paradox. In the case of the Cretan liar,

the statement \underline{a} can have the SNS states D or X and then

the input and the output of the equation $\underline{a} N = \underline{a}$ are the same,

and there is no difficulty. In simple language, the former (namely the state D) means that we cannot say whether what the Cretan says is in fact true or false; a distinction between the two states T and F is not possible. This is one solution. Another solution is to say that the statement is invalid and impossible. Such a statement cannot be included in a logical argument, for it is neither true nor not true.

The discrimination whether the first solution (namely that $\underline{a} = D$) or the second solution (namely $\underline{a} = X$) should be considered as the actual solution of the equation $\underline{a} \sim = \underline{a}$ depends upon the particular problem, and would depend on semantic considerations, rather than considerations of pure logic. Thus, if \underline{a} is the statement "This proposition is not true", then, the state D adds nothing to the knowledge in that argument. On the other hand, it can be considered invalid, and rejected from the argument, by the consideration that it leads to the contradictory state X for \underline{a} .

(d) The state D having semantic value-properties of infinity

As mentioned in Section 6(a), there may be situations where the doubtful state $D (\equiv T \text{ or } F)$ for the truth value of the solution of the paradoxical statement $\underline{a} \Rightarrow \neg \underline{a}$ may contribute to the understanding of the semantic consequences of this statement. Thus, if we talk of the integer "infinity" (∞), then the question whether it is odd or even cannot be answered definitely either way; and the answer to the question: "Is ∞ odd?" is that it may be so, or it may not be so. This is because we have the well-known equation (40) satisfied by ∞ (given, for instance, in [8], p. 86):

$$\infty = \infty + 1 \quad (40)$$

Because of this equation (40) which ∞ obeys, we obtain that if ∞ is odd then it is even, and if it is even then it is odd.

This has the structure of the CLP, namely $\underline{a} \Rightarrow \neg \underline{a}$. Therefore, from purely logical considerations, as discussed in Section 5 and 6,

we can have either of two possible answers to the question:

"Is ∞ odd or even?" namely

(a) The question about the odd or even nature of infinity is inadmissible (state X), (41a)

(b) Infinity can be either odd or even and it is not possible to decide between these two possibilities (state D). (41b)

Which of these is the real answer depends upon mathematical theory, and not on logic. The author believes that the latter is a more reasonable statement for the even-odd nature of infinity, as deduced from the equation $\infty = \infty + 1$. In fact, several more properties of ∞ can be seen to have the D state, if this state is included along with the states T and F in sentential logic.

For example, for any pair of finite numbers x and y, either of the relations $x > y$ or $x \leq y$ holds; but one excludes the other. Hence, if we denote the statement $x > y$ by \underline{t} , then

\underline{t} is either T or F, for finite numbers. However, it follows that \underline{t} may be "T or F" ($\equiv D$) if $y = \infty$ and $x = \infty + 1$, so that the answer to the question "Is $x > y?$ " is that it may be "yes" and may be "no". Hence, questions, for which a definite answer having one of the states T or F is possible for finite numbers, may have only an answer in the state D when the numbers become infinite. A study of the considerations regarding the need and relevance of the SNS state D in dealing with such properties of ∞ will be taken up for detailed study in a later report. Some new results are obtained regarding cardinal numbers of infinite sets.

7. Absence of Paradoxes and Incompleteness in Multi-valued Logic

- (a) Impossibility of incompleteness due to closure of Boolean algebras.

Just as we have shown that BA-2 and BA-3 are closed with respect to matrix connectives (either unary or binary; and in the forward or reverse form) and the Boolean connectives \oplus and \otimes in the forward direction, it is obvious, from the general theory of BA-n in Part I [2], that this is the case for all BA-n generated by n basic states. Hence, closure is assured for all n-valued logics, having n independent states ($n \geq 2$), leading to a set of 2^n possible states for that logic. Thus, the application of the theory in the earlier sections to BA-n, assures the absence of incompleteness (as is commonly understood) in any multi-valued logic.

- (b) Non-closure of a new type leading to higher genus Boolean algebras

However, reversal of Boolean connective relations in BA-n can lead to a number of new states not contained in BA-n. Thus,

in BA-3, the reversal of the equation in (42a) leads to (42b):

$$(\mathcal{V} \otimes \exists = \exists) \equiv (\exists \overset{\leftarrow}{\vee} \exists = \mathcal{V}) \quad (42a,b)$$

This has as solution " \exists " or Δ , but not Φ ", which is not contained in BA-3. If we take BA-7, having for its seven basic states the seven non-impossible states of EPL (BA-3), then this will be contained in it. Thus, we need BA-m with m = $(2^7 - 1)$ for representing all possible states generated by solutions of equations of the type (40b) in EPL.

This process is recursive, and, as mentioned in [1] and [3], there is an essential incompleteness in the formulation of mathematical logic. Thus, a logic, based on BA-n, necessarily leads to one based on BA-m, with m = $2^n - 1$, and so on ad infinitum. However, we have not been able to find examples of arguments that require more than EPL, isomorphic to BA-3.

(c) Absence of paradoxes in SNS and EPL and in all logic with n basic states ($n \geq 2$)

The impossibility of incompleteness in logics isomorphic to BA-n ($n \geq 2$) is also indicative of the result that no paradoxes can be formulated in them. Thus, considering SNS, we have already seen that the typical pair of paradoxical statements as in DSP, namely $\underline{\underline{a}} \sim \underline{\underline{b}}$, $\underline{\underline{b}} \sim \underline{\underline{a}}$ (or the CLP $\underline{\underline{a}} \sim \underline{\underline{a}}$), has the permissible state D as a solution for $\underline{\underline{a}}$. This is possible because $T \oplus T^C = T \oplus F = D$ is one of the admissible states of SNS.

In exactly the same way, if we have the following equations in EPL —

$$\underline{\underline{a}} \approx^{E^C} (i, i^C) = \underline{\underline{b}} ; \quad \underline{\underline{b}} \approx^{E^C} = \underline{\underline{a}} \quad (43a, b)$$

leading to

$$\underline{\underline{a}} \approx^{E^C} (i, i^C) = \underline{\underline{a}} \quad (44)$$

we will find that, both for (43) and for (44), if $\underline{\underline{a}_1}$ is the input and $\underline{\underline{a}_2}$ the output state for $\underline{\underline{a}}$, then

$$\text{For } \underline{\underline{a}_1} = q_i, \underline{\underline{a}_2} = q_i^C ; \quad \underline{\underline{a}_1} \vee \underline{\underline{a}_2} = \emptyset \quad (45)$$

so that the two complementary states q_i and q_i^c lead to a contradiction, similar to what happens with T and F in the DSP and the CLP of PC. However, it is readily seen that

$(q_i \oplus q_i^c) \equiv \Delta$ is a permissible solution of (43), or (44).

Since Δ is included in EPL itself, there is no paradox produced by (45a,b).

Even more generally, we can say that, because EPL is closed as described in Section 7(a) above, no set of relations formulated in EPL can lead to a state not contained in it. Hence, no paradox will be generated by an argument in EPL (and in SNS). The same line of argument will give the still more general result that no logic isomorphic to the Boolean algebra BA-n (which is closed under the conditions mentioned above) can lead to a paradox.

The only exceptions are those obtained by the reversal of relations having Boolean connectives, when there is absence of closure and a higher genus Boolean algebra is required to describe and represent the result of the argument.

Thus, in full generality, we can say that mathematical logic, employing Boolean algebraic vectors and matrices, is a complete story, having no inconsistencies or paradoxes.

References

1. Ramachandran, G.N., Syad-Nyaya System (SNS) — A New Formulation of Sentential Logic and its Isomorphism with Boolean Algebra of Genus 2, Current Science, 1982, 51, 625-636.
2. Ramachandran, G.N., Vector-Matrix Representation of Boolean Algebras and Application to Extended Predicate Logic (EPL) Part I, Current Science, 1983, 52, 292-302.
3. -do- Part II, Current Science, 1983, 52, 335-341.
4. -do- Part III "Algebraic Theory for Categorical Statements in Quantified Predicate Logic (QPL)" (Submitted for publication) (Matphil Reports No. 34).
5. -do- Part IV "General Theory for Statements of Type-1 in QPL". (Matphil Reports No. 35, Draft 2.)
6. -do- Part V ^a"Theory for/ general argument in logic including SNS, QPL and QSN statements" (Matphil Reports No. 36, Draft 2).
7. Stoll, R. R., "Set Theory and Logic", W.H. Freeman and Company, New York, 1963.
8. Russell, B., "Introduction to Mathematical Philosophy", Simon and Schuster, New York, 1982.

TABLES

	<u>Page No.</u>
1. SNS truth table for the vidya operator <u>V</u> .	18
2. Reverse relations obtained from Table 1.	18

FIGURES

1. Graph of the essential logic of the Double Statement Paradox	
(a) By breaking the circularity at <u>a</u>	34
(b) By breaking the circularity at <u>b</u>	34
2. Logical graph of the contradictory statements in	
(a) the CSP paradox as given in Eqs. (26a,b,c)	38
(b) the CSP-2, as in (30a,b,c,d).	38
3. Logical graph of the Cretan Liar Paradox	
(a) Circular argument as in Eqn. (27)	40
(b) With circularity broken and introducing the vidya operator	40

Molecular Biophysics and Biomedical Research*

(Acceptance Talk at the award ceremony of the
Rameshwardas Birla Award for medical and related
fields on 28th January 1985)

G.N. Ramachandran

INSA Albert Einstein Professor
Mathematical Philosophy Group
Indian Institute of Science
Bangalore 560 012

*Matphil Reports No. 38, December 1984

Molecular Biophysics and Biomedical Research*

(Acceptance Talk at the award ceremony of the
Rameshwardas Birla Award for medical and related
fields on 28th January 1985)

G.N. Ramachandran

INSA Albert Einstein Professor
Mathematical Philosophy Group
Indian Institute of Science
Bangalore 560 012

Matphil Reports No. 38, December 1984

Molecular Biophysics and Biomedical Research*

G.N. Ramachandran
INSA Albert Einstein Professor
Mathematical Philosophy Group
Indian Institute of Science
Bangalore 560 012

I would like to say, first of all, how deeply touched I was when I learnt that the Rameshwardas Birla Award for Medicine has been given to me for 1984-85. It is a great honour to be chosen for this Award, but it is particularly so because this is the first time that the Committee has selected a person who is not working on practical aspects of medical science, but only in basic fields related to biology and medicine, namely molecular biophysics and molecular biology.

I shall not try to give a complete outline of the achievements in the two laboratories devoted to molecular biophysics which I have established in the University of Madras, and in the Indian Institute of Science, respectively. However, I shall touch upon some aspects of these studies in which I was personally connected quite closely. One of the earliest among these was the enunciation of the triple helical structure of

*Matphil Reports No. 38, Dec 1984 (Acceptance Talk at the award ceremony of the Rameshwardas Birla Award for medical and related fields on 28th January 1985).

.2.

collagen, in 1954, very soon after the structure of alpha helix was discovered from theory by Pauling in 1951, and the DNA double helix by Watson and Crick in 1953. The most essential feature of the chemistry of collagen which led to the discovery of the triple helix is shown in Slide 1*, namely its amino acid composition which consists of one third glycine (the smallest of all amino acids) and approximately another one third of the two amino acids proline and hydroxyproline, with rigid side chains. These features had not been specifically incorporated in the design of several structures made previously by very eminent scientists in the early 1950s. The structure which we proposed is shown from the side in Slide 2. The three helices run parallel to one another and are interconnected by hydrogen bonds. The architecture is made clearer by looking down the triple helix from the top down the central axis and the backbone of the three helices is shown in Slide 3. In this, each helix has a three-fold screw axis and the three of them are also related by a three-fold screw axis interconnecting them. However, the x-ray diffraction pattern of collagen, shown in

*The slides will be shown in the lecture, but the diagrams are not included in the written report.

.3.

Slide 4, indicated that the number of units per turn is not 3 but 3.3. This was adopted for the triple helix and then the pattern of the winding of the three helices about a common central axis was discovered to be as shown in Slide 5. Essentially, the structure of collagen could be said to have been solved when this pattern of the backbones in the three individual helices was worked out. These studies were made during 1954-55 and the essential correctness of the structure is shown by an optical diffractometer picture, of a drawing of the collagen structure, which is shown in Slide 6. The occurrence of a strong meridional reflection on the 10th layer and off-meridional reflections in the 3rd and 7th and the weak ones on the 4th, completely agreed with the x-ray pattern shown earlier and they justify an approximate value of 3.3 for the number of residues per turn. Afterwards, it was a question of fitting various pieces of information regarding the protein into the structure, and the ball and spokes model shown in Slide 7 (which was the best structure available in 1968) was obtained in our laboratory. (It should be mentioned that although the occurrence of the triple helix with 3 residues

per turn was first worked out by us in 1954, the supercoiled triple helix with 3.3 residues per turn was arrived at simultaneously, both in Madras, and in London and Cambridge in 1955).

The spurt of activity in biophysics and molecular biophysics that came out after the discovery of the essential helical nature of many biopolymers and also the discovery of the crystal structures of globular proteins which came at about the same time, led to the recognition that biological function is completely explicable from the molecular structure and chemical interactions of the component biochemical entities in a living system. Consequently, one could make experiments in the laboratory on aspects of various biological systems and try to find out the exact mechanism of the reactions that take place in normal biological activity and in pathological conditions. Many studies of this type in biochemistry and molecular biology were facilitated to a great extent by the understanding of the chemical structure and molecular architecture of biomolecules obtained by using theoretical techniques as well as various methods of biophysical chemistry.

In the evolution of the theoretical techniques of molecular biophysics, workers in our laboratory in Madras had contributed to a material extent. The first consistent explanation of the principles governing the conformation, or three dimensional structure, of a protein molecule was theoretically worked out in Madras in 1963, using a pair of peptide units as the model system. The picture of this is shown in Fig.8 and two dihedral angles (rotation angles ϕ and ψ) are marked in this diagram. It may be shown theoretically that the pair of angles (ϕ , ψ) can occur only for certain combinations of these and we worked out which of these are allowed and which of these are disallowed, by employing contact criteria of rigid, rubber-like spherical atoms. The resultant diagram first published in the Journal of Molecular Biology, 1963, has come to be known now-a-days as the 'Ramachandran Diagram' and is shown in Slide 9. This slide is taken from a well-known text book on Biochemistry by Lehninger which was published as early as 1975 and is used widely in graduate and post-graduate teaching in USA. It is understood that now-a-days this diagram is taught even to high-school students in biochemistry.

.6.

Very soon after this diagram came into being, the corresponding energy diagram was worked out in several laboratories including ours, and a typical one is shown in Slide 10. It shows regions of (ϕ, ψ) which are most likely to occur, and regions which are forbidden. It is very reassuring to see that when a large number of protein structures are examined, the diagram is fully substantiated, as may be seen from Slide 11 which is taken from a recent book on protein structure published from Germany.

Soon after the enunciation of the (ϕ, ψ) -rotation angles in proteins, the essential rotation angles that are necessary for the specification of the three dimensional structures of nucleic acids and polysaccharides were also developed in Madras (during 1965-70), although they were also worked out in other laboratories during the same period. The diagram showing the dihedral angles required for a nucleotide chain, worked out in Madras, is shown in Slide 12 and the corresponding diagrams for two saccharide units in α -glucose and β -glucose

are also shown in Slides 13 and 14. The essential difference between the stable configurations of the latter two is very clear, namely that the polymer of α -glucose (amylose) is going round and round while the chain is rigid and straight in the polymer (cellulose) of β -glucose. Amylose is nothing but starch and cellulose is the essential component of wood. So the difference in the properties of food and wood is to be attributed to this molecular mechanism occurring for the linking of the saccharide units that go to build up the molecular chains of these. It should be mentioned that V. Sasisekharan was associated with me in working out the dihedral angles of the nucleotide chain, and similarly, V.S.R. Rao for polysaccharide chains, in our laboratory in Madras. Both of them have gone ahead and built up schools of research of their own in these two respective subjects, during the last twenty years mainly in Bangalore.

We shall not discuss the various steps by which the branch of molecular biophysics has interacted with molecular biology in its development. However, during the last few years,

scientists in the latter field have even succeeded in imitating Nature in the production of new living organisms. Biotechnology and bioengineering are highly specialized experimental fields covering aspects of biology that were almost unthought of even twenty years ago. The confidence with which scientists working in these fields go towards new directions is not to a small extent the consequence of the great spurt in the knowledge of biomolecules created by biophysical chemistry and the theoretical and physical methods that were developed since the 1950's and 60's.

As a consequence, we could imitate chemicals that produce specific actions in living systems and produce them in the laboratory for being given to persons who lack in such chemicals, due to genetic reasons, or otherwise. But this is only a small part of the great achievements of modern biochemical research. Now-a-days, we can even improve upon Nature. By studying in minute detail the biochemistry and molecular structure and action of the component chemicals in a particular tissue or organism and trying to give this a good theoretical explanation

from ideas of physical chemistry and chemical physics, it has become possible to design drugs that are even more effective than the chemicals that have been chosen by Nature to carry out the particular function. Thus, molecular biophysics has played a subtle but important part in the explosive developments in molecular biology during the last decade.

A good example is the explanation of the mode of action of the well-known antibiotic penicillin. This drug acts on the bacterial cell wall whose molecular structure is shown in Slide 15. It consists of a series of parallel polysaccharide chains which are linked together by short peptide chains in which the sequence D-Ala-D-Ala occurs. The structure of penicillin is shown in Slide 16 and, looked at in a careful manner, it has a sequence of L-Val-D-Cys. The chemists were always puzzled by the fact that the D-D sequence is imitated by the L-D sequence of the peptide chain, which is chemically the cause for the inhibition of the synthesis of the peptide chains required to build the bacterial cell wall. In a study made in Bangalore by Prof. V.S.R. Rao and colleagues, they could

show very clearly that the theoretically allowed conformations of D-Ala-D-Ala which is free and of L-Val-D-Cys in the beta-lactam configuration in penicillin are, in fact, very closely similar to one another. The best conformation of each of these has been pictured and shown side by side in Slide 17 and a close similarity in the locations of the biochemically important groups in the two molecules is very clearly seen. Thus we have, for the first time, a purely physicochemical explanation of the mode of action of penicillin. Starting from this, the workers in Bangalore have shown why certain drugs of the penicillin group are more effective than others, and so on, and studies of this type should help very much in the design of the new drugs of the penicillin type which could have the required properties.

We shall give one more example concerning the biochemistry of vitamin C, where pure theory, combined with ideas from different fields of sciences, have led to results of relevance to medicine. Our studies on collagen are closely related to

this and led to the proof of Pauling's theory that vitamin C increases resistance to diseases. This discovery arose out of pure curiosity on my part in trying to understand the molecular structure of a component known as complement C1q of the immune response system, in comparison with the triple-helical structure of the individual molecules of collagen. Collagen fibres require hydroxyproline for the stability of its triple helical structure, and hydroxyproline is not taken as such from the pool of amino acids and incorporated in the molecular chain of collagen that is being built up; rather proline is incorporated at first and then, where it is needed, the proline side groups are hydroxylated by an enzyme which produces hydroxyproline. The way in which hydroxyproline stabilizes the interaction between two neighbouring helices in the triple helix is shown in Slide 18. It will be seen that a water molecule occurs in the hydrogen-bonds linking an NH group of one chain and the C=O group of a neighbouring chain and the same water molecule also links with the OH group of hydroxyproline, thus forming a tight interconnecting link between the two neighbouring helices to make the triple helix very stable.

Detailed study of the chemical process of hydroxylation made by several persons, including, in particular, Prockop in USA, indicated that the enzyme requires as co-factor ascorbic acid (vitamin C) for the mechanism to be carried out. At about the same period, namely the late 1960's and the early 1970's, the molecular structure of complement C1q had also been investigated and elaborated in the laboratory of Porter in Oxford, U.K. One of the features that came out of these studies was that complement has plenty of hydroxyproline, and also contains one third glycine as in collagen, so that it should possess a triple-helical structure similar to collagen. From electron microscope studies, Porter and collaborators could in fact show the existence of the triple helix in its molecular architecture. The two features — triple helix and occurrence of hydroxyproline — previously known to occur only for collagen, was thus delineated in complement C1q. Discussing about such medico-chemical inter-relationships with Prof. George Pickering, former Dean of Medicine in Oxford, I was rudely shaken when he insisted that the suggestion of Linus Pauling that vitamin C

is not only needed for good health, but that it also improves immunity to diseases, was all "Bosh". In a flash the association of the words Pickering-Oxford-Porter-complement led to the ideas of hydroxyproline-collagen triple helix-Prockop-vitamin C, and I could see, from these that vitamin C (ascorbic acid) is as much required for the synthesis of the collagen-like filaments of C1q as for collagen itself. Thus, the exact possibility whereby vitamin C improves immunity became clear by analogy — complement requires ascorbic acid for production, and Pauling's suggestion is absolutely sound and theoretically explicable.

The logical interconnections between these ideas which were known to everybody (including me) for a number of years, is shown in Slide 19. However, the remarkable conclusion mentioned above had to be brought out by the stimulus of personal discussions. I believe that it was the interaction with senior scientists in various fields, which was made possible for me, by being provided with the very prestigious Fogarty International Fellowship by US National Institutes of Health (which gave me a whole year of freedom in 1977 to talk with

learned scientists all over the United States) that led to the conclusions shown in the slide.

I have many suggestions for the creation of a proper atmosphere in the institutions of higher learning in our country which will really provide interdisciplinary exchange of thoughts. Due to lack of time, I shall only mention one, namely that, just as biochemistry flowered as a discipline for study in Indian Universities and institutions in the first half of this century, biophysics, which really grew very fast in the second half, must also be included as a separate discipline in every teaching and research institution dealing with bio-sciences at the master's degree level and in higher stages of learning. Somehow, the progress in establishing full-fledged courses in biophysics has been slow in development, and it is to be hoped that rapid progress in this direction will come in the next few years.
